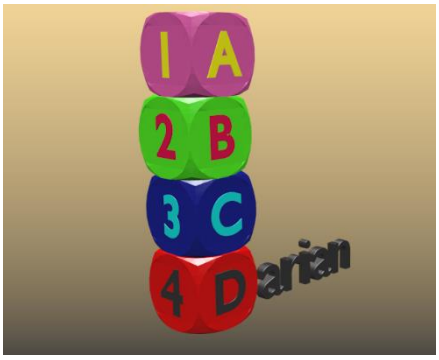
HTL | MÖSSINGERSTRASSE

HÖHERE TECHNISCHE BUNDESLEHRANSTALT
KLAGENFURT, MÖSSINGERSTRASSE

ABTEILUNG FÜR ELEKTRONIK UND TECHNISCHE INFORMATIK

DIPLOMARBEIT

ABC-Darian



eingereicht von Vanessa Sereinig
Andreas Zammernig

Projektbetreuer Dipl.-Ing. Dr. Helmut Jeremias

Diese Diplomarbeit entspricht den Standards gemäß Leitfaden für Diplomarbeiten des BM|BF.

Klagenfurt, am .....

HTL | MÖSSINGERSTRASSE

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe.

Klagenfurt, am 08.04.2015

Verfasser/innen:

Vanessa Sereinig

Andreas Zammernig

1 Kurzbeschreibung des Projektes

1.1 Kurzfassung

1.1.1 Aufgabenstellung (Assignment of Tasks)

Legasthenie (Lese-/Rechtschreibschwäche) und Dyskalkulie (Rechenschwäche) sind für betroffene Personen meist sehr unangenehm und können bis zur sozialen Ausgrenzung führen. Ein frühzeitiges Erkennen dieser Störungen und deren Ursachen ist für eine erfolgreiche Gegensteuerung unbedingt notwendig.

Im Zuge unseres Projektes wird ein Lernprogramm für Kinder und Jugendliche entwickelt, das spielerisch räumliches Denkvermögen schult. Die Verwendung der professionellen Entwicklungsumgebung Unity3D garantiert ein Design nach aktuellem Stand der Technik.

Das Programm soll spielerisch beim Lernen helfen und mehrere, immer schwieriger werdende, Level umfassen, sodass die Lernschwäche des Kindes Schritt für Schritt zurückgeht.

1.1.2 Realisierung (Implementation)

- Maximaler Lernfortschritt durch optimierte Behandlungsmethoden
- Gamedesign in Unity3D (Scripts, Objekte, Animationen)
- Interaktive und kindgerechte Spielumgebung
- 3D-Objekterstellung in Blender
- Einbindung der KINECT zur Gestensteuerung
- Benutzerfreundliches Interface
- Datensicherungsprinzip für mehrere Spielstände
- Verifizierung und Test der einzelnen Level

1.1.3 Ergebnisse (Results)

Konzept

3D-Modellierung, GUI-Design, Gestenerkennung

Planungsunterlagen

Klassenentwurf, C#-Scripting, KINECT, Algorithmenumsetzung

Realisierung

Testläufe, Bedienungsanleitung, Erweiterungsmöglichkeiten

1.2 Abstract

Many children suffer from dyscalculia or dyslexia. These learning disorders make the lives of those kids, their parents and other relatives very difficult. Even the easiest exercises demand a great deal of the concerned ones. Determining learning disorders in an early state, as well as taking therapies and special working lessons is very important to improve the healing process.

ABC-Darian is the perfect addition to the expensive private therapies. It's a game that helps these children to get over their disorders and offers them a playful alternation to their usual boring learning process. And this can all be done at any time, at any place.

To guarantee that the game is promising, ABC-Darian works in a three-dimensional (3D) environment. Requiring the spatial thinking is a proven method to increase the success of an exercise. The 3D objects that are necessary to create the surroundings are designed in the modelling program Blender. The programing work is done in Unity3D, a game engine which allows the programmer to make up a 3D and also 2D world. The language used is c#.

To also show the user how much he or she already accomplished, the game includes high score lists for all levels and overall recording of the reached points and achievements.

In order to get the absolute attention of the kid while playing, the Kinect is added to some of the levels. This way it is possible for the users to interact in the game by using their gestures.

The combination of the 3D objects and the innovative game control offers a whole new way for children to unconsciously improve their skills.

ABC-Darian will, of course, also be designed in a child-like and unique style all over the levels. Exercising the brain was never so much fun.

Inhaltsverzeichnis

1 Kurzbeschreibung des Projektes 5
1.1 Kurzfassung 5
1.1.1 Aufgabenstellung (Assignment of Tasks) 5
1.1.2 Realisierung (Implementation) 5
1.1.3 Ergebnisse (Results) 5
1.2 Abstract 6
2 Einleitung 9
3 Grundlagen und Methoden 11
3.1 Lernschwächen 11
3.1.1 Verschiedene Formen 11
3.1.2 Die AFS-Methode 12
3.2 Gestensteuerung 13
3.3 Unity3D 13
3.3.1 Lights 14
3.3.2 MonoDevelop 14
3.4 Asset Store 14
3.5 Blender 14
3.5.1 Exportieren von 3D-Modellen 14
3.6 Scripting 15
3.6.1 C#-Scripting 15
4 Individuelle Zielsetzung der Teammitglieder 19
4.1 Teammitglied „Vanessa Sereinig“ 19
4.1.1 Aufgabenstellung 19
4.1.2 Grundlagen und Methoden 20
4.1.3 Realisierung 31
4.1.4 Ergebnisse 52
4.2 Teammitglied „Andreas Zammernig“ 61
4.2.1 Aufgabenstellung 61
4.2.2 Grundlagen und Methoden 62
4.2.3 Realisierung 73
4.2.4 Ergebnisse 81
5 Zusammenfassung 87
6 Literatur und/oder Quellenverzeichnis 89

6.1 Allgemeines....89
6.2 Teammitglied: Sereinig....89
6.3 Teammitglied: Zammernig....90
7 Anhang (Dokumente und Protokolle)....91
7.1 Projektdokumentation....91
7.1.1 Pflichtenheft....91
7.1.2 Systemstruktur....93
7.1.3 Projektstruktur....94
7.1.4 Risikoabschätzung....95
7.1.5 Terminplan....96
7.2 Arbeitszeitnachweise....97
7.3 Technische Dokumentation....99
7.3.1 Unity3D....99
7.3.2 Blender....100
7.3.3 Kinect....101
7.4 Arbeitspakete....103
7.5 Businessplan....114
7.5.1 Executive Summary....114
7.5.2 Ist-Analyse (Produkt, Umfeld, Unternehmen, Team)....115
7.5.3 Marketingplan....118
7.5.4 Projektplan....122
7.5.5 Finanzplan (1. Geschäftsjahr)....123
7.5.6 Business Model Canvas....126
7.5.7 Literaturverzeichnis....127
7.6 Abbildungsverzeichnis....127

2 Einleitung

In Österreich gibt es 129.759 Kinder, die an Legasthenie oder Dyskalkulie leiden. Dies entspricht einem Anteil von 15% in Bezug auf die Anzahl der hier lebenden Kinder unter 14 Jahren. [STAT 15]

Zur Behandlung dieser Lernschwächen werden kostenintensive Therapien angeboten, um den Kindern den Schulalltag etwas zu erleichtern und mit ihnen ein bis zwei Mal die Woche Übungen durchzuführen. Doch besonders wichtig beim Lernen in diesem jungen Alter sind vor allem der Spaß an der Sache, sowie eine gute Erreichbarkeit der Unterrichtsressourcen für eigenständiges Üben zu Hause. Auch Abwechslung spielt eine große Rolle für den Erfolg von solchen Therapien – sobald Langeweile eintritt, verliert das Kind sofort das Interesse und das Lernen wird zur Qual.

ABC-Darian ist ein Programm, das Kindern mit Lernschwäche die Möglichkeit bietet, spielerisch an ihren Fähigkeiten zu arbeiten und diese unterbewusst zu verbessern. Es soll vor allem eine Ergänzung zu den üblichen Behandlungsmethoden darstellen und Kindern sowie Eltern eine angenehme Abwechslung zu den Schwierigkeiten beim Bewältigen des Lern- und Schulalltags bieten.

Für optimalen Lernerfolg wird ein Großteil der einzelnen Levels in einer dreidimensionalen Umgebung gestaltet. Wenn beim Übungsprozess das räumliche Denkvermögen mit einbezogen wird, ist dies ein bewiesener Vorteil für effizientes Lernen. Zusätzlich soll der/die BenutzerIn insofern gefordert werden, dass er sich auch interaktiv mit Gesten am Spielhergang beteiligen kann. Um dies zu ermöglichen, wird die KINECT in das Programm eingebunden und in einigen Levels zur Steuerung dienen. Durch diese besonderen Eigenschaften hebt sich ABC-Darian von anderen Spielen dieser Art ab und ist somit ein effizientes Lernspiel, das Spaß macht.

Im Zuge dieses Projekts sollen diverse Minispiele entwickelt, die benötigten 3D-Objekte modelliert und die grafische Oberfläche kindgerecht und spannend gestaltet werden.

3 Grundlagen und Methoden

3.1 Lernschwächen

Bereits im Kindergartenalter entwickeln sich beim Menschen sowohl ein Grundverständnis von Zahlen und Mengen, als auch eine grobe Fähigkeit zur Abschätzungen von Längen- und Flächeneinheiten. In den ersten Schuljahren werden diese Kenntnisse dann ausgebaut. Es werden die mathematischen Grundrechenarten, beginnend mit einfachen Additionen, erlernt und anschließend wird sich intensiver mit dem Zählen beschäftigt. Auch ein Gespür für logische Abläufe und Verknüpfungen werden entwickelt. Dieser Lernprozess beinhaltet viele verschiedene Schritte, welche auf die vorhergehenden aufbauen und sich in der Schwierigkeit sukzessive steigern. [DYS 01]

3.1.1 Verschiedene Formen

Es gibt mehrere Beeinträchtigungen, die einem jungen Menschen das problemlose Erlernen von gewissen Fähigkeiten nehmen, sogenannte Lernschwächen.

Die wohl bekanntesten sind:

- **Lese-Rechtschreibschwäche (Legasthenie)**
  Siehe 4.2.2.1
- **HKS (hyperkinetisches Syndrom)**
  Den Betroffenen fällt es schwer, ihre Aufmerksamkeit für längere Zeit auf ein und denselben Vorgang zu richten und still sitzen zu bleiben. Die Hyperaktivität drückt sich darin aus, dass das Kind nicht still sitzen kann, sehr impulsiv reagiert und Schwierigkeiten bei der Konzentration hat.
- **Rechenschwäche (Dyskalkulie)**
  Siehe 4.1.2.1

Oft fällt eine Diagnose einer solchen Lernschwäche, vor allem bei HKS, äußert schwer, da eine Unterscheidung zwischen krankhaftem und normalem (kindischen) Verhalten meist sehr schwer ist. Ob das Kind aus purem Trotz nicht verstehen will oder wirklich nicht kann, muss genauestens untersucht und getestet werden. [DYS 02]

3.1.2 Die AFS-Methode

Abbildung 3-1: Die AFS-Methode

Die AFS-Methode (in Abbildung 3-1 grafisch dargestellt) ist ein pädagogisch-didaktischer Förderansatz für Menschen mit Legasthenie und Dyskalkulie. Sie legt ihren Schwerpunkt auf das multisensorische Lernen, d.h., es werden alle Sinne gezielt in den Lernvorgang eingebaut. Es gibt drei grundlegende Bausteine, auf denen die AFS-Methode aufgebaut ist:

- Verbesserung der Aufmerksamkeit beim Rechnen, Schreiben und Lesen:
  - Da dyskalkule und legasthene Menschen Probleme dabei haben, die Aufmerksamkeit gezielt auf Zahlen bzw. Worte zu lenken und dort zu halten, ist das Erlernen der Aufmerksamkeitsfokussierung unbedingt notwendig, um die Lernschwäche abbauen zu können.
- Verbesserung der Sinneswahrnehmungsleistungen (Funktion):
  - Das Funktionieren der Basissinne stellt einen wesentlichen Faktor beim Erlernen des Rechnens dar. Therapien mit akustischen und optischen Elementen, sowie der Förderung des räumlichen Denkvermögens, sind daher von großem Lernerfolg gekrönt.
- Verbesserung der Rechenleistungen (Symptom):
  - Die Verbesserung auf der Symptomebene ist für den Gesamterfolg natürlich von enormer Bedeutung. Dabei ist darauf zu achten, dass auf die jeweiligen Bedürfnisse der Betroffenen Rücksicht genommen wird. Ein Training muss also individuell und gezielt erfolgen, auch in diesem Bereich. [AFS 01]

3.2 Gestensteuerung

Sehr viele Unternehmen fassen die Möglichkeit mit natürlichen Körpergesten zu interagieren sehr gut auf. Viele Firmen investieren in die Entwicklung der neuen Art der Steuerung. Diese Art der Kommunikation ist eine Neuheit und wird schon in vielen Bereichen von Spielen bis hin zu per Hand dimmbaren Leuchten eingesetzt.

Das Besondere an der Gestensteuerung ist die Interaktivität und die Innovation. Dabei wird der gesamte Körper oder nur ein gewisser Bereich des Körpers, wie zum Beispiel die Arme der bedienenden Person, erfasst und bei der Ausführung einer definierten Bewegung - also einer Geste - wird ein Ereignis ausgelöst. Ein Beispiel könnte das Regeln der Helligkeit einer Leuchte sein. So wird beim nach-links-Wischen mit der Hand die Leuchte heller und beim nach-rechts-Wischen wird sie dunkler.

Um eine Geste erkennen zu können, wird eine entsprechende Hardware benötigt. Sie muss die Person erfassen und seine Bewegungen wahrnehmen können. Die Hardware muss aber auch tolerant sehen und nicht auch die kleinste Bewegung schon als Geste zu identifizieren, da der menschliche Körper nie völlig still steht. Sie sollte aber immer die Gesten erkennen, egal von welcher Person sie ausgeführt werden. Das heißt, es darf keine Rolle spielen, ob eine Person klein, groß, schmal oder etwas breiter ist.

Als Beispiel für solch eine Hardware wären die Kinect und die Leap motion.

3.3 Unity3D

Unity3D ist eine Spiel-Engine, mit welcher Spiele und interaktive Anwendungen im zwei- und dreidimensionalen Bereich erstellt werden können. Unity3D selbst stammt von dem Unternehmen Unity3D Technologies, welches seinen Sitz in San Francisco in den USA hat. Die Entwicklungsumgebung kann kostenlos heruntergeladen und verwendet werden. Parallel zur kostenfreien Basisversion existiert eine Vollversion, die sich durch einige zusätzliche Funktionen von der Basisversion unterscheidet. Die Zielplattformen reichen vom PC über Spielkonsolen bis hin zu Smartphones.

Um in Unity3D eine Anwendung zu erstellen, werden eine oder mehrere Szenen erstellt. Es wird immer jeweils eine behandelt, welche dann im Hauptfenster dargestellt wird Um eine Szene mit Leben zu erfüllen, können beliebig viele Game-Objekte und 3D-Modelle, wie zum Beispiel die Kamera oder eine Spielfigur, per Drag-and-Drop hinein gezogen werden. Diese wiederum können nach Belieben durch das Hinzufügen von Skripts, Texturen, Sound, Animationen und weiteren Komponenten manipuliert werden. Hat man dann eine solche Anwendung fertig entwickelt, kann diese für die gewünschte Zielplattform durch die Entwicklungsumgebung erstellen.

Für die Erstellung eines Game-Objektes steht von Unity3D bereits eine Vielzahl von primitiven Grundkörpern zur Verfügung: Cube, Sphere, Capsule, Cylinder, Plane, Quad.

3.3.1 Lights

Lights dienen, wie es der Name bereits sagt, um Licht in die Szenen von Unity3D Licht zu bringen. So können mit ihnen einzelne Objekte oder die ganze Szene beleuchtet werden. Je nach eingesetztem Light und der Positionierung werfen Objekte auch Schatten.

Will man eigene Objekte haben, welche zum Beispiel eine andere Form haben, so müssen diese durch das Zusammenfügen mehrerer primitiver Grundkörper erzeugt werden oder in einem externen Programm zur Objektmodellierung wie Blender kreiert und anschließend exportiert und in Unity3D importiert werden.

3.3.2 MonoDevelop

Die Programmierung innerhalb eines Skripts ist in folgenden Skriptsprachen möglich: UnityScript (vergleichbar mit JavaScript), C# und Boo.

Bei der Installation von Unity3D kann die Entwicklungsumgebung MonoDevelop mitinstalliert werden. MonoDevelop unterstützt den Entwickler bei dem Erstellen von Skripts. Es kann aber alternativ auch jede andere Entwicklungsumgebung, welches die verwendete Skriptsprache unterstützt, zum Schreiben der Skripte herangezogen werden.

3.4 Asset Store

Im Unity3D Asset Store können sogenannte Packages erworben werden. Packages können zum Beispiel ein Game-Objekt enthalten, welches mit einem Skript eine besondere Funktion erfüllt. Es können aber auch ganze Szenen in solchen Packages heruntergeladen werden. Der Großteil der zu Verfügung stehenden Inhalte kostet etwas, es gibt aber auch welche die gratis sind. [UNI 14]

3.5 Blender

Blender ist eine 3D-Grafiksoftware, welche Funktionen zum Modellieren dreidimensionaler Körper enthält. Neben dem geometrischen Verändern der Objekte ist es auch möglich sie zu texturieren, zu animieren und zu rendern. Auch ein Videoschnitteditor und ein Spiel-Engine sind in Blender enthalten, sodass auch das Erstellen von Games (Skriptsprache: Python) und das Simulieren von physikalischen Eigenschaften (Flüssigkeiten, Schatten,…) möglich ist.

3.5.1 Exportieren von 3D-Modellen

Um Objekte, die in Blender modelliert wurden, auch in Unity3D verwenden zu können, gibt es eine Auswahl von zu exportierenden Dateiformaten in Blender:

- .fbx

- .dae (Collada)
- .3DS
- .dxf
- .obj [EXP 01]

3.5.1.1 Vorteile

- Nur die Daten, die man exportieren will, werden exportiert
- Überprüfbare Daten
- Kleinere Files
- Unterscheidung in Teilobjekte (jedem Teilobjekte können andere Komponenten zugewiesen werden)
- Unterstützung von anderen 3D-Formaten

3.5.1.2 Nachteile

- Möglicherweise langsamer beim Erstellen eines Prototypen
- Möglicherweise Verlust der Übersicht über Versionen der verschiedenen Quellen und Gaming Daten (z.B. fbx-files)

3.6 Scripting

In Unity3D kann mit verschiedenen Scripting-Sprachen gearbeitet werden:

- Boo
- Java Script (Js)
- C Sharp (C#, Cs)

Für ABC-Darian wird die Scripting-Sprache C# verwendet.

Skripts können in Unity3D als sogenannte Komponenten auf Objekte angewandt werden. Sie beschreiben bestimmte Abläufe und arbeiten quasi parallel. Es können mehrere Skripts an einem einzelnen Objekt hängen und auf jedes kann in allen anderen Skripts dieses Objekts mit dem Befehl „this.GetComponent<SkriptName>()“ zugegriffen werden. Sie können z.B. aus- und eingeschaltet werden oder auf bestimmte Variablen oder Methoden anderer Skripts zugreifen, sofern diese Variablen entsprechend deklariert sind.

3.6.1 C#-Scripting

3.6.1.1Grundgerüst

In Abbildung 3-2 sieht man den groben Aufbau eines C#-Skripts für Unity3D.

```
using UnityEngine;
using System.Collections;

public class MainPlayer : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {

    }
}
```

Abbildung 3-2: Aufbau eines Skripts in Unity3D

3.6.1.1.1 Methoden

- **void Start()**
  Diese Methode wird zu Anfang des Skriptsaufrufs durchlaufen. In ihr werden Startbedingungen gesetzt und generelle Szeneneinstellungen durchgeführt. Sie kann nicht wieder aufgerufen werden. Wenn es Code-Stellen gibt, die sowohl in der Start() als auch in anderen Programmteilen aufgerufen werden sollen, so sollte man diese in eigene Methode schreiben, die dann an den jeweiligen Passagen aufgerufen werden.

- **void Update()**
  Diese Methode wird in jedem Frame des Spieleablaufs aufgerufen (Frameanzahl hängt von der sogenannten Framerate ab, welche in fps angegeben wird: Frames per Seconds). In ihr sollten laufende Überprüfungen realisiert und Variablen für die weitere Verarbeitung von Ereignissen gesetzt werden (z.B. Überprüfung, on eine Taste gedrückt wurde).

3.6.1.1.2 Weitere vordefinierte Methoden:

- **void OnGUI()**
  In ihr kann das grafische User Interface, welches von der Kamera angezeigt wird, manipuliert werden.

- **void OnApplicationEnd()**
  Wird aufgerufen, sobald das Skript bzw. die Szene beendet wird, bzw. kurz davor. Ist sehr empfehlenswert wenn es um das Wiederherstellen von Grundeinstellungen geht, wie das Wiedereinschalten der Sichtbarkeit des Cursors.

- **void OnTriggerEnter(), void OnTriggerExit()**
  Wird aufgerufen, sobald ein Trigger ausgelöst bzw. wieder „verlassen“ wird. Dies ist praktisch zur Erkennung der Kollision zweier Objekte bzw. der Reaktion darauf (das getroffene Objekt muss hierfür einen Collider besitzen, dessen Trigger gesetzt ist). Das Erkennen des Verlassens eines Triggers kann sinnvoll sein, wenn ein Spieler sich nur

in einem bestimmten Bereich aufhalten soll. Verlässt er diesen – er berührt ihn nicht mehr – so wird ein TriggerExit ausgelöst und es kann darauf reagiert werden.

4 Individuelle Zielsetzung der Teammitglieder

4.1 Teammitglied „Vanessa Sereinig“

4.1.1 Aufgabenstellung

Pflichtenheft:

- Erstellen des Pflichtenheftes gemeinsam mit dem Projektpartner

Auswahl und Kennenlernen der Entwicklungsumgebung:

- Einlesen in die Programme „Unity3D“ und „Blender“.
- Abwägen der nutzbaren Techniken zum Erstellen der Spiele unter Berücksichtigung der recherchierten Behandlungsmöglichkeiten.

Erstellung User Interface:

- Gestaltung des GUI in Unity3D.
- Geforderte User Interfaces:
- Einstellungen und Optionen, Levelauswahl, Anzeige über den Lernfortschritt, Startmenü (Charakterauswahl,…), usw.

Levelentwicklung:

- Gestaltung der 2D/3D-Welt, in der sich die Third Person bewegt.
- Auswahl der optimalsten Programmiermethode zum Erstellen und Einbinden der verschiedenen Blender-Objekte.
- Methode der Spielstandspeicherung auswählen.

Gemeinsame Levelgestaltung:

- Zusammen mit dem Projektpartner die erforderlichen Objekte (Spielcharaktere, Gegenstände, Umgebung) besprechen und diese eigenständig in Blender kreieren.
- Absprache des generellen Levelaufbaus gemeinsam mit dem Projektpartner, um eine individuelle Einzelarbeit mit anschließender Kombination zu erreichen.

Recherche zu Dyskalkulie:

Anstellung diverser Recherchen zu dem Thema Dyskalkulie.
Die Nachforschung sollte beinhalten:

- Behandlungsmöglichkeiten
- auffälligste und häufigste Ausprägungen v. Dyskalkulie
- beliebte Lernspiele
- Optimaler Umfang und Schwierigkeitsgrad der einzubindenden Rechenschritte

Dokumentation:

- Dokumentation lt. Anforderung der HTL Mössingerstraße

4.1.2 Grundlagen und Methoden

4.1.2.1 Lernschwäche: Dyskalkulie

4.1.2.1.1 Beschreibung und Definition

Dyskalkulie, auch Zahlenblindheit, Rechenschwäche oder Arithmasthenie genannt, ist eine Beeinträchtigung des arithmetischen Denkens. Nicht zu verwechseln damit ist der sogenannte Zahlenanalphabetismus, bei dem es den Betroffenen schwer fällt, in Zahlen dargestellte Sachverhalte zu verstehen. [DYS 03]

Leidet ein Kind unter Dyskalkulie, so erschwert dies den Lernprozess erheblich. Schon die einfachsten logischen Aufgaben sind kaum oder gar nicht zu bewältigen. Das Einschätzen von Mengen, das Ordnen von Zahlen nach ihrem Wert und auch das Zählen fallen den Betroffenen unglaublich schwer. Ihnen fehlt die nötige Basis um die Grundrechenarten erlernen zu können. Da diese Personen Zahlen einzig als Symbole verstehen, und nicht als Mengenangaben, fehlt ihnen das wesentliche Verständnis um Lernschritte in der Mathematik zu verinnerlichen. [DYS 01]

Oftmals zeigen sich keinerlei Verhaltensauffälligkeiten. Jedoch bei bestimmten schulischen Anforderungen, aber auch anderen Lebensbereichen, die etwas mit visuellem Gedächtnis, Zeit- und Raumorientierung, Ordnungsstrukturen, Organisation und Handlungsabläufen zu tun haben, gibt es oft enorme Schwierigkeiten. Diese können im Einzelfall sehr schwere Auswirkungen erreichen. [DYS 02]

4.1.2.1.2 Merkmale

Ein Kind, das an Dyskalkulie leidet, wird…

- ungewöhnlich lange zum Bewältigen von einfachen Berechnungen brauchen,
- Probleme beim Unterscheiden und Einschätzen von Mengen haben,
- die Zahlensymbole zwar mit deren Namen in Verbindung bringen können, jedoch den Wert als solches nicht verinnerlichen,
- Schwierigkeiten beim räumlichen Denkvermögen haben,
- schon den Zahlenraum 1-20 als große Herausforderung ansehen.
- bei jeder Lösung fragen, ob diese richtig ist,
- den Rechenweg nicht logisch nachvollziehbar erklären können,
- Rechenschritte, die es gestern scheinbar konnte, am nächsten Tag vergessen haben,
- beim Einmaleins immer wieder die Reihen von unten hoch zählen. [DYS 02]

4.1.2.1.3 Unterschiede und Besonderheiten

Dyskalkulie sagt nichts über die Intelligenz des Betroffenen aus. Ein überdurchschnittlicher IQ ist bei ihnen nicht selten zu finden. Die Fähigkeit zur Beweisführung in der höheren Mathematik

muss nicht zwingend beeinträchtigt sein, da arithmetische und mathematische Fähigkeiten nicht zwangsläufig miteinander verknüpft sind. Dass ein Dyskalkuliker mit einer Begabung abstrakte Probleme in der höheren Mathematik lösen kann, obwohl die einfachsten arithmetischen Aufgaben (wie Grundrechenschritte mit natürlichen Zahlen) für ihn nur schwer zu bewältigen sind, ist nicht ungewöhnlich. [DYS 02]

4.1.2.1.4 Statistik

Eine Dyskalkulie tritt bei fünf bis sieben Prozent der Weltbevölkerung auf (gemessen an Kindern und Jugendlichen zwischen 6 und 15 Jahren). Im Schnitt sitzt damit in jeder Klasse ein Dyskalkuliker, der ein „Genügend“ als Jahresnote in Mathematik nur mit sehr viel Mühe mit Hilfe der Eltern und Nachhilfelehrern schaffen kann. Trotz der enormen Leistungen, die für diese Schulnote notwendig sind, kann eine vier in Mathematik einem sonst passablen Schüler schnell die Zukunft verbauen. Förderungen, wie eigene Betreuer, vom Land gibt es zwar, aber zu wenig. Häufig werden Förderschulen als alternative vorgeschlagen, diese sind aber auf Grund der Tatsache, dass die Schwäche allein in Mathematik sitzt, eine Unterforderung für das Kind. [DYS 04]

4.1.2.1.5 Behandlung/Therapie

Kinder mit Dyskalkulie brauchen spezifische Förderung von Experten, nur so kann sie auch erfolgreich behandelt werden. Voraussetzung ist eine genaue klinisch-psychologische Abklärung und die Erstellung eines individuellen Förderprofils. Es gibt nicht nur einen richtigen Behandlungsansatz, sondern jedes Kind braucht ein speziell abgestimmtes Therapiekonzept. [DYS 05]

Inhalte der Therapie

Es ist nicht nur von großer Bedeutung eine stete Übung in das Leben des Kindes einfließen zu lassen, sondern dies muss auch mit einigen anderen Faktoren verknüpft werden. Rechenübungen allein und der daraus entstehende Mangel an Selbstwertgefühl können sonst zu einem erheblichen Problem in der Psyche des Kindes führen.
Die drei Ebenen, die bewältigt werden müssen, sind also: Defizite in Mathe, veränderte Denkweise im Lernen und psychische Reaktion auf den andauernden Misserfolg.

In der Therapie werden vor allem folgende Bereiche gefördert:

- Wahrnehmungsleistung (akustisches und visuelles Training, Gedächtnistraining etc.)
- Konzentration
- Rechenkompetenzen (z.B. Mengenverständnis, Textaufgaben, Zahlenvorstellung, Zehnerüberschreitung etc.)
- Vermittlung von Lerntechniken[DYS 05]

Ziele der Therapie

Ziel einer Therapie ist die deutliche Verbesserung der Rechenleistung der Kinder und die Lust und Freude am Lernen zurückzubringen. Zusätzlich ist es von großer Wichtigkeit, dass mögliche psychische Begleitsymptome mitbehandelt werden.
Mit klinisch-psychologischen Methoden kann eine Verbesserung in folgenden Problemfeldern erzielt werden:

- Umgang mit Prüfungs-, Schul- und Versagensängsten
- Steigerung des Selbstwertgefühls
- Erhöhung der Motivation
- Linderung der psychosomatischen Beschwerden
- Besserung von Verhaltensauffälligkeiten[DYS 05]

4.1.2.2 Objekterstellung – Blender

4.1.2.2.1 Allgemeines zu Blender

Abbildung 4-1: Blender Logo

Blender ist eine frei verfügbare 3D-Grafiksoftware. Neben der Hauptfunktion, nämlich dreidimensionale Körper zu modellieren, ist auch möglich diese zu texturieren, zu animieren und zu rendern. Sie wurde 1994 von der Firma Blender Foundations entwickelt. Teilweise ist das Programm Blender auch in Deutsch erhältlich.

Die Skriptsprache ist Python. Blender enthält sowohl einen Videoschnitteditor also auch eine Spiel Engine. Somit lassen sich kleinere Programme generieren, aber auch Firmenlogos oder Videoclips erstellen. [BLEN 01]

Blender im Einsatz

Hier werden einige aktuelle Beispiele für eine aktive und globale Verwendung von Blender angeführt:

- **Orange Movie Project:**

Im Mai 2006 war die Premiere dieses Filmprojektes. Einige der angesehensten Blender Künstler und Entwickler produzierten im Laufe dieses Projektes einen Kurzfilm, den sie ausschließlich in freier Software programmierten und renderten.

- **Big Bucks Bunny:**

Ein weiteres Filmprojekt, welches am 20. Mai 2008 auf DVD erschien. Es stellt das Folgeprojekt von Orange Movie Project dar. In Abbildung 4-2 kann ein Ausschnitt des Projekts betrachtet werden.

Abbildung 4-2: Big Bucks Bunny

- **Open Movie Projekte:**

Weitere Film-Projekte wurden über die Jahre hin mit Blender und bestritten, wie diverse Open Movie Projekte. Im Oktober 2011 startete das vierte *Open Movie Project* der Blender Foundation mit dem Projektnamen Mango. Ziel des Projekts war es, die Funktionalität im Bereich visueller Effekte zu erweitern (siehe Abbildung 4-3). Der fertige Film trägt den Namen „Tears of Steel“. [BLEN 01]

Abbildung 4-3: Einsames Haus mit Rendering

4.1.2.2.2 Funktionalität

Blender arbeitet mit Polygonnetzen, NURBS-Oberflächen, Metaball- und Textobjekten.

NURBS-Oberflächen = Non uniform rational B-Splines

Das sind mathematisch definierte Kurve oder Flächen, die zur Modellierung beliebiger Formen verwendet werden. Jede beliebige Linie, bzw. jeder Linienzug, kann mit NURBS dargestellt werden, solange sich diese nicht verzweigen.

Metaball

Ein Metaball ist das Ergebnis eines Algorithmus, der eine dehnbare Oberfläche erzeugt. Diese nimmt bei genau einem Metaball die Form einer Kugel an, bzw. bei mehreren eine Menge von ineinander gehenden Kugeln. Der Algorithmus geht auf Jim Blinn in den 1980er Jahren zurück (Veranschaulichung, siehe Abbildung 4-4: Zwei Metabälle verschmelzen zu einem/ein Metaball wird von einem zweiten eingedrückt). [BLEN 01]

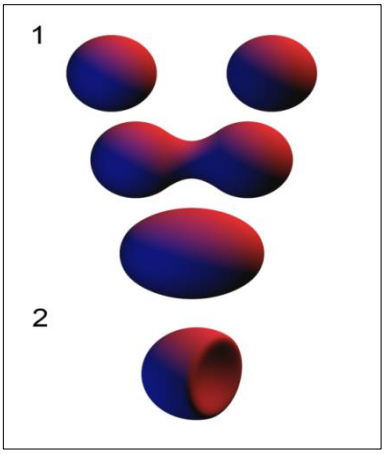


Abbildung 4-4: Transormation zweier Metabälle

4.1.2.2.3 Wichtige Funktionen in Blender

Funktionen, die zur optimalen Modellierung gebraucht werden:

- Unterstützung für die Bearbeitung von Polygonnetzen:
    - Unterteilungsflächen (Subdivision Surfaces)
    - Multiresolution-Netze, also Objekte in mehreren Auflösungsstufen
    - Sculpting-Werkzeuge, mit denen Objekte wie virtueller Ton verformt werden können.
- Material- und Texturierungswerkzeuge:
    - Verschiedene UV-Unwrapping-Methoden (u.a. Angle-Based, Conformal, Smart Projections)
    - knotenbasierte Material- und Textureditoren, die sehr komplexe Materialien erlauben.
    - 3D-Malwerkzeuge erlauben das Malen direkt auf dem Modell
    - Grafiktablettunterstützung

Funktionen, die für das Projekt nicht gebraucht werden:

Diese Funktionen sind für ABC-Darian weniger von Belang, da Blender nicht als 3D-Spieleengine verwendet wird, sondern lediglich zu Objektmodellierung.

- Animationswerkzeuge:
    - Shape-Keys (Morph Targets)
    - Charakteranimationen mit direkter und inverser Kinematik
    - Kurven- und Lattice (Käfig)-basierte Verformungen
    - Simulation elastischer Objekte (Softbodies) einschließlich Kollisionserkennung
    - Simulation von Festkörpern (Rigidbodies), ebenfalls mit Kollisionserkennung
    - Kleidungssimulation

  - partikelbasierte Haare, Gräser usw.
  - Partikel mit Kollisionserkennung
  - Flüssigkeitsdynamik

- Multithreadfähige interne Renderer:

  - Kantenrendern (Cartoon-Linien)
  - Raytracing
  - Umgebungsverdeckung (Ambient Occlusion)
  - Radiosity
  - mehrschichtiges Rendering und Compositing

- 3D-Spieleengine

- Videoschnittprogramm

Kompatibilität mit Unity3D

Blender benutzt ein eigenes mit *.blend* gekennzeichnetes Dateiformat, auf welches mit anderen Programmen nur schwer zugegriffen werden kann. Es ist plattformübergreifend und auf- und abwärtskompatibel. Import- und Exportskripte sind für verschiedene Formate integriert, z. B. für OBJ- und 3DS-Dateien.

Das Dateiformat, welches für unser Projekt mit Unity3D ausschlaggebend ist, ist **Collada (.col)**, mit welchem man ein Objekt als solches exportieren, in Unity3D importieren und dort als GameObject verwenden kann. Sie haben die gleichen Bestandteile wie in Blender (z.B. licht, Kamera, Plane,…), welche je nach Einstellung der Modellierung in Blender einzeln bearbeitet (z.B. gefärbt und vergrößert) werden können.

4.1.2.2.4 Arbeitsmodi

In Blender gibt es mehrere Arbeitsmodi, mit denen man das eingefügte Objekt bearbeiten kann. Es ist unter anderem möglich es zu skalieren, seine Form zu verändern, es auf jegliche Weise zu erweitern, es optisch (mit Texturen oder Bemalungen) anzupassen und seine Position zu verändern. Da es vor allem wichtig ist, wirklich nur das abzuändern, auf das es ankommt, sind einige abgesicherte Arbeitsbereiche notwendig. Jeder dieser Bereiche – oder Modi – besitzt spezielle Werkzeuge um bestimmte Aufgabe erledigen zu können, ohne dabei auf andere Ebenen des Objektes zuzugreifen.

Es kommt bei der Objektsmodellierung also einerseits auf das geschützte Arbeiten in abgetrennten Bereichen an, andererseits auf den problemlosen Wechsel zwischen diesen.

Die meisten Objektmodellierungsprogramme verwenden, um diese Differenzierung zu erreichen, eine Auswahl an verschiedensten Werkzeugen. Blender hingegen bietet ganz klar voneinander getrennte Arbeitsmodi an. Dieser kontextbezogene Arbeitsansatz ist für den Benutzer ein klarer Vorteil für die Übersichtlichkeit. [BLEN 03]

Folgende Modi gibt es in Blender:

- Object Mode
- Edit Mode

- Sculpt Mode
- Vertex Paint
- Texture Paint
- Weight Paint

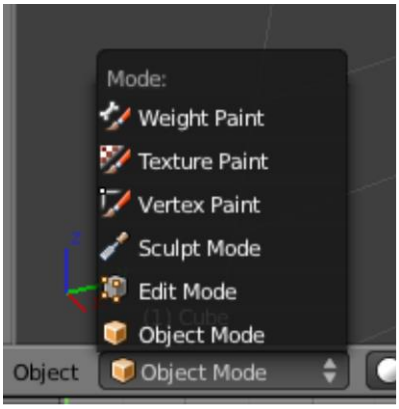


Abbildung 4-5: Modemenü in Blender

Standartmäßig befinden sich alle Objekte entweder im Object- oder Edit-Mode. Mit „tab“ kann zwischen ihnen gewechselt werden. Um in die anderen, spezielleren Modes zu gelangen, benutzt man am besten das Menü (siehe Abbildung 4-5), da die Shortcuts hierfür wenig selbsterklärend sind.

Folgende Modi sind jene, welche für das Projekt essentiell sind:

Object Mode

Dies ist – neben dem Edit Mode – der am häufigsten verwendete Modus. Wird ein Objekt neu erstellt bzw. eingefügt, so ist es zu Beginn standartmäßig in diesem Arbeitsbereich. Das Objekt kann hier bewegt, gedreht und skaliert werden. Die Hauptaufgaben sind das Erstellen von Gruppen, das Bearbeiten von Child/Parent-Beziehungen und das richtige Platzieren der einzelnen Teilobjekte. [BLEN 04]

Edit Mode

Der Edit Mode dient der detaillierten Modellierung von einem Objekt. Grundsätzlich dient er der Manipulation der Geometrie. Es kann jeder Teil des Objekts u.a. gezerrt, gelöscht, dupliziert, verformt oder rotiert werden. Für unser Projekt ist dieser Mode ausschlaggebend, da Blender wirklich nur für die Objektmodellierung genutzt wird. Für ein kindgerechtes Design ist es daher wichtig, eigene, und vor allem einheitliche, Formen (Umgebungsdetails,…) für Kinder zu kreieren.

Um zu modellieren kann folgende, grundlegende Auswahl zwischen den „Teilen“ eines Objekts getroffen werden:

- Vertex Select (Punkt/e, Ecke/n selektieren)
- Face Select (Fläche/n selektieren)
- Edge Select (Kante/n selektieren)

Mit verschiedenen Shortcuts (siehe Tabelle 4-1: Liste der essentiellen Shortcuts und Hotkeys in Blender) können mehrere Elemente gleicher Art selektiert werden. Benutzt man ein Tool,

während mehrere Flächen/Ecken/Kanten markiert sind, so wird es auf alle ausgewählten angewendet. [BLEN 04]

In Abbildung 4-6 kann man einen kleinen Ausschnitt des Arbeitstoolmenüs im Edit-Mode sehen.

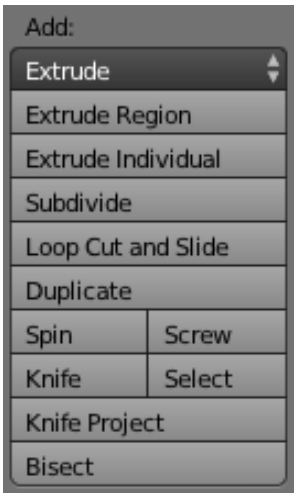


Abbildung 4-6: Tools im Edit-Mode

- **Extrude**: Erweitert die selektierte Region/die selektierten Einzelteile in die durch Ziehen oder Tastendruck (erst Achse, dann Zahlenwert, der die Stärke der Erweiterung angibt) angegebene Richtung und Stärke.
- **Subdivide:** Teil die selektierten Teile des Objektes in mehrere, kleinere Stücke desselben Typs auf. Vor allem bei Faces sehr nützlich. Jede selektierte Fläche wird in vier Teilflächen unterteilt.
- **Dublicate:** Kopiert den selektierten Teil des Objekts, auch wenn es sich dabei nur um eine Ecke handelt.

4.1.2.2.5 Objekttypen

- **Meshes**
  Meshes sind Objekte, die aus polygonalen Flächen, Kanten und/oder Eckpunkten bestehen. In Blender können diese mit diversen Bearbeitungstools editiert werden. Auch das zusammensetzen mehrerer Meshes ist möglich, um mit vielen verschiedenen Formen ein Ganzes zu schaffen.
- **Kurve**
  Kurven (Curves) sind mathematisch definierte Objekte, die mit „handles“ oder Kontrollpunkten manipuliert werden können (Kurvenverlauf, -skalierung,…).

- **Oberflächen**
  Oberflächen (Surfaces) sind vier- oder mehrseitige Flächen, die über Kontrollpunkte manipuliert werden. Sind speziell für simple Flächen sehr nützlich, vor allem wenn es um das Abrunden geht.
- **Metabälle**
  Metabälle (Metas) sind Objekte, die durch eine mathematische Funktion geformt werden. Das Volumen (dreidimensional), in dem dieses Objekt existiert, wird von dieser Funktion definiert. Vorstellen kann man sich Metabälle als Tropfen, welche flüssigkeitsähnliche Eigenschaften annehmen, sobald man zwei oder mehr Metabälle verwendet.

- **Text (**)**
  Textobjekte (Texts) erzeugen eine 2D-Darstellung einer Zeichenkette. Obwohl sie zweidimensional sind, können sie trotzdem skaliert, rotiert und in der Position verändert werden.
- **Armatures**
  Skelette (Armatures) werden verwendet, um 3D-Modelle so auszurüsten (riggen), dass sie animiert werden können.
- **Empty (*)**
  Empties sind leere Objekte, die nicht mit gerendert werden. Sie werden hauptsächlich dazu verwendet, um die Position oder Bewegung anderer Objekte zu kontrollieren, ohne jedoch gesehen zu werden (unsichtbare Wände, um die Bewegungsfreiheit einzuschränken).
- **Kameras(**)**
  Die Kamera legt den Bildausschnitt fest und speichert diesen. Es können mehrere Kameraobjekte festgelegt werden.
- **Lampen (**)**
  Lampen (Lighting) werden verwendet, um Lichtquellen in der Szene zu platzieren. Es gibt mehrere Arten, wie das Licht fallen kann, zum Beispiel Directional Light (beleuchtet die gesamte Richtung, in die es strahlt, großflächig), Point Light (sehr punktierter, dünner Lichtstrahl in die eingestellte Richtung)
- **Kraftfelder (*)**
  Kraftfelder (Force Fields) simulieren externe Kräfte (Gravitation, Wind, usw.). Sehr hilfreich sind diese in physikalischen Simulationen, oder wenn es darum geht, Bewegungen realitätsnah darzustellen.

(*) Wird für das Projekt nicht von Gebrauch sein.
(**) Wird nur für das Logo/das Banner verwendet, ansonsten für das Projekt uninteressant. [BLEN 04]

4.1.2.2.6 Shortcuts und Hotkeys

Folgende Shortcuts sind die für die in 2.3 genannten Funktionen in Blender zuständig. Ein Arbeiten ohne die Verwendung der entsprechenden Tastenkombinationen wäre sehr umständlich, da es den Modellierungsprozess durch Suchen der entsprechenden Werkzeuge im Menü unnötig verlangsamen würde.

Funktion	Shortcut (ab Version 2.65)	Mode
... nur auf der Y- und Z-Achse	SHIFT+X	Object und Edit
... nur auf der X- und Z-Achse	SHIFT+Y	
... nur auf der X- und Y-Achse	SHIFT+Z	
Skalieren (*)	S	
Rotieren (*)	R	
Alles auswählen	A	

Mehrere auswählen	SHIFT + Rechtsklick	
Markierungskreis nutzen	C	
… vergrößern	Mausrad	
Koordinaten öffnen/schließen	N	
Aktion Rückgängig machen	STRG+Z	
Aktion wiederherstellen	STRG+SHIFT+Z	
Objektflächen ausblenden (nur Kanten)	Z	
Mesh Tools öffnen/schließen	T	
Pivotpunkt neu setzen	STRG+ALT+SHIFT+C	
Curser verschieben	Linksklick	Object
Objekt auswählen	Rechtsklick	
Objekte auswählen	SHIFT+Rechtsklick	
Objekt frei verschieben (*)	G	
Curve/Text/Meta zu Mesh (und v.v.)	ALT + C	
Objekte vereinen	STRG+J	
Objekt(e) löschen	ENTF	
Objekt(e) duplizieren	SHIFT+D	
Curser verschieben	Linksklick	Edit
Punkt/Kante/Fläche auswählen	SHIFT+Rechtsklick bzw. Rechtsklick	
Punkt/Kante/Fläche frei verschieben(*)	G	
Specials öffnen	W	
Punkt/Kante/Fläche/(n) löschen	ENTF	
Punkt/Kante/Fläche/(n) duplizieren	SHIFT+D	
Mehrfachauswahlwerkzeug (Rechteck)	B	
Merhfachauswahlwerkzeug (Kreis)	C	
UV Mapping Menü öffnen	U	
Normals außen/innen neu kalkulieren	STRG+SHIFT+N*	

Tabelle 4-1: Liste der essentiellen Shortcuts und Hotkeys in Blender

(*) Bei diesen Funktionen ist es möglich, auch eine Achse zu bestimmen, auf die es angewendet werden soll. Dies geschieht mit einem dem Shortcut folgenden Tastendruck von X, Y oder Z. [KEY 01]

4.1.2.3 Persistente Datenspeicherung

4.1.2.3.1 File IO

Das Scripting via C#-Files macht es möglich, Daten in Files zu schreiben und aus ihnen zu lesen. Dies geschieht ganz einfach mit Input- und Outputstreams. Mit der Klasse File können neue Files erstellt werden, welche unter den Assets abgespeichert werden (Pfad kann angegeben werden).

Beispiel

```
private void btnWrite_Click(object sender, EventArgs e)
{
    string file_name = "C:\\test1.txt";

    System.IO.StreamWriter objWriter;
    objWriter = new System.IO.StreamWriter(file_name);

    objWriter.Write(textBox1.Text);
    objWriter.Close();

    MessageBox.Show("Wrote File");
}
```

Abbildung 4-7: Beispiel für File IO

Wie in Abbildung 4-7 dargestellt, kann man mit File-IO Klassen Zeichenketten ganz einfach abspeichern. In diesem Beispiel wird beim Betätigen eines Buttons der Inhalt einer Textbox in ein File geschrieben. Die verwendete Streaming-Klasse ist System.IO.StreamWriter, welcher das Speichern von beliebig langes Strings bzw. Character-Sequenzen ermöglicht.

Wie man sieht, ist es zum Schluss wichtig, den StreamWriter wieder zu schließen, da es sonst zu einer Exception kommt. [FIO 01]

4.1.2.3.2 PlayerPrefs

PlayerPrefs ist eine Klasse in Unity3D, die es einem ermöglicht ohne viel Aufwand Daten persistent zu speichern. [PP 01]

Funktionen

Funktion	Beschreibung
DeleteAll	Löscht alle Daten der PlayerPrefs
DeleteKey	Löscht einen Key und seine zugeordneten Daten
GetFloat	Gibt den Wert, der einem bestimmten Key zugeordnet ist, zurück, wenn vorhanden
GetInt	Gibt den Wert, der einem bestimmten Key zugeordnet ist, zurück, wenn vorhanden
GetString	Gibt den Wert, der einem bestimmten Key zugeordnet ist, zurück, wenn vorhanden
HasKey	Gibt “true” zurück, wenn ein gewisser Key vorhanden ist
Save	Schreibt alle vorgenommenen Änderungen auf die Festplatte
SetFloat	Setzt den Wert der Präferenz, die mittels Key identifiziert wird
SetInt	Setzt den Wert der Präferenz, die mittels Key identifiziert wird
SetString	Setzt den Wert der Präferenz, die mittels Key identifiziert wird

Tabelle 4-2: Funktionen von PlayerPrefs

4.1.3 Realisierung

4.1.3.1 Generelles

4.1.3.1.1 Minispiel verlassen

Wenn man ein Minispiel verlassen und zum Startbildschirm zurückkehren möchte, muss man „ESC“ drücken, woraufhin in der unteren (meist rechten) Ecke ein neuer Button erscheint, der wie ein Notausgang-Schild aussieht. Auf den muss geklickt werden. Bei Betätigung wird der Benutzer/die Benutzerin gefragt, ob er/sie sich sicher ist und wirklich beenden möchte. Bei „Ja“ wird der Spielstand gespeichert und der Levelauswahl-Bildschirm erscheint, bei „Nein“ geht das Spiel an der Stelle weiter, an der man unterbrochen hat.

Code

- In der Update()-Methode wird abgefragt, ob die ESC-Taste gedrückt wurde. Wenn ja, wird die boolesche Variable „esc“ auf true gesetzt.

```
if(Input.GetKeyDown(KeyCode.Escape)){
    esc = true;
}
```

- In der OnGUI()-Methode wird daraufhin wegen der Variable „esc“ in der if()-Anweisung der Button angezeigt, welcher in der unteren rechten Ecke platziert und mit der

entsprechenden Textur versehen wird. Wird er gedrückt, so wird ein Thread (eine Co-Routine) aufgerufen.

```
//--- exit Button
GUIStyle exitStyle = new GUIStyle ();
if (esc) {
      //Screen.showCursor = true;
      if (GUI.Button (new Rect (w - 75, h - 75, 50, 50),
         exitTexture, exitStyle)) {
            StartCoroutine (getBackHome ());
      }
}
```

- Die Co-Routine „getBackHome()“ warten erst eine Sekunde und ruft danach den Levelauswahl-Bildschirm auf. Hierfür wird der Befehl *Application.LoadLevel(<Szenenindex>)* verwendet (Szenenindex ist die Stelle, an der die jeweilige Szene im Built-Verzeichnis gereiht ist).

```
IEnumerator getBackHome ()
{
      yield return new WaitForSeconds (1f);
      Application.LoadLevel (1);
}
```

Dieser Verlassen-Modus wird in allen Minispielen verwendet.

4.1.3.1.2 Instruktionsbildschirm

Der Instruktionsbildschirm wird vor jedem Level angezeigt und beschreibt mit Bildern und Text, was das Ziel des Spiels ist und wie es zu steuern ist. Da ABC-Darian vorwiegend für Kinder im Grundschulalter entwickelt wurde, wird der angezeigte Text ebenso vorgelesen, sodass es nicht zwingend nötig ist, dass eine Aufsichtsperson dem Kind das Spiel erklären muss. Es soll so selbsterklärend und verständlich wie möglich sein.

Szenenaufbau

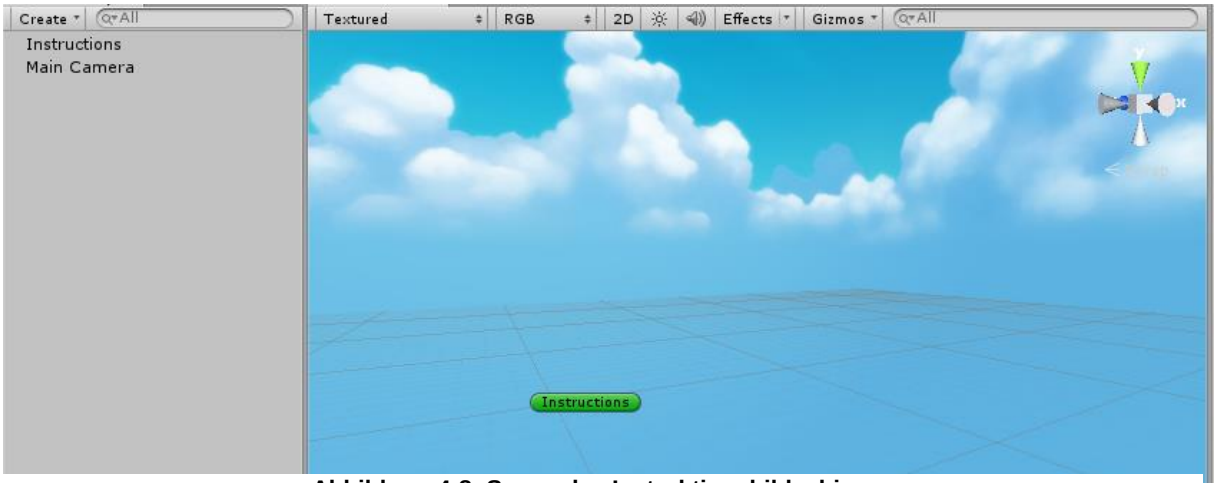


Abbildung 4-8: Scene des Instruktionsbildschirms

In der Szene (siehe Abbildung 4-8) befinden sich zwei Objekte:

- Main Camera

- Instruktionsset (Instructions)

  Das Instruktionsset ist ein an sich leeres Objekt, welches nur die Komponenten *Transform* (Skalierung, Position und Drehung des Objekts) und das Skript *InstructionGUI* besitzt.

Außerdem ist ihr eine Skybox zugewiesen (Tageslicht-Himmel).

Code

```
switch (nextLvl) {

case 1: //evaluate biggest
    heading = "Apfelpyramiden";
    subHeading = "Auf welchen der beiden Seiten sind mehr Aepfel?";
    instrTxt = "Entscheide, welche Pyramide aus mehr Aepfeln besteht.\n\nTipp:
Abzaehlen";
    instrHeading1 = "Klicke auf eine Seite:";
    texture_instr1 = texturesInstr [0];
    guiOrderNumber = 2;
    break;
case 2: //memory
    heading = "Memory";
    subHeading = "Finde die Paare!";
    instrTxt = "Finde zu jeder Rechnung das passende Ergebnis. Jedes Ergebnis kommt
nur einmal vor.";
    instrHeading1 = "Klicke auf die Karten:";
    texture_instr1 = texturesInstr [0];
    instrHeading2 = "Beispiel:";
    texture_instr2 = texturesEG [0]; //2nd instruction is replaced by example-texture
    guiOrderNumber = 3;
    break;
case 3: //count cubes
...
```

Dieser Code steht in der OnStart()-Methode, da er gleich zu Beginn wichtig ist. Er beschreibt die switch-Anweisung, die je nach darauffolgendem Level die jeweiligen Variablen für die OnGUI() befüllt. Die switching-Variable „nextLvl“ ist ein Integerwert, in dem der Index des nächsten Levels im Builtmenü gespeichert ist. Für jedes Minispiel gibt es einen switch-case.

4.1.3.1.3 Startmenü

Beim Starten der Szene wird mit der Veränderung des Kamerazooms ein Effekt erzeugt, der dem Benutzer das Gefühl gibt, auf die GUI hinzufliegen. Die GUI besteht aus fünf Buttons.

Code

Hier sieht man die Anzeige der Hauptbuttons und was passiert, wenn sie gedrückt werden. Der Code steht in der OnGUI()-Methode.

```
if (zoomingEnded) {
        GUILayout.BeginArea (new Rect (w / 3 + w / 20, h/4,
        w / 3 - w / 10, h / 2));
        if (GUILayout.Button ("Starte neu", buttonStyle)) {
            setParameters (true, false, false, false);
        }
```

```
        if (GUILayout.Button ("Laden", buttonStyle)) {
            setParameters (false, true, false, false);
        }
        if (GUILayout.Button ("Einstellungen", buttonStyle)) {
            setParameters (false, false, false, true);
        }
        if (GUILayout.Button ("Über", buttonStyle)) {
            setParameters (false, false, false, false);
        }
        if (GUILayout.Button ("Beenden", buttonStyle)) {
            setParameters (false, false, true, false);
        }
        GUILayout.EndArea ();
    }
```

Die Methode setParameters() ist dazu da, um bestimmte boolesche Variablen zu setzen, was nötig ist, um in der Update immer abzufragen, welche gerade true ist. Ist eine Variable gesetzt, so wird darauf (meist mit dem Starten eines anderen Skripts und dem Schließen des aktuellen) reagiert.

```
    private void setParameters (bool start, bool load, bool end, bool set)
        {
            this.load = load;
            this.end = end;
            this.start = start;
            this.set = set;
        }
```

4.1.3.2 Modellierte Objekte

Folgende Objekte wurden für das Projekt in Blender entworfen und modelliert. Zur optimalen Verwendung wurden alle Objekte so entworfen, dass jedes Einzelteil auch in Unity3D verändert werden kann (mit dem Join-Befehl können diese Teilobjekte miteinander verbunden werden, sodass sie zu einem Objekt verschmelzen). Generell kann auf jedes dieser Teile individuell zugegriffen werden, was aber meist nicht nötig sein wird. Im Normalfall wird eher das gesamte Objekt in der Szene skaliert und platziert.

Die Farben, mit denen die Objekte in Blender eingefärbt werden, sind Materialien, die auf die entsprechenden Flächen angewandt werden und eine gewisse Grundfarbe besitzen. Meist weist ein Objekt mehrere Farbmaterialien auf.

Vor dem Exportieren ist es wichtig, Kameras und Lampen zu entfernen, da diese in Unity3D unnötig für das Objekt sind. Das Dateiformat sollte *.fbx* oder *.col* sein, diese können in Unity3D optimal verwendet werden. Ausnahme hierbei bilden nur 4.1.3.2.1 („Logo“) und 0 (Banner), da bei diesen Objekten nur das gerenderte Bild als *.png*- oder *.jpg*-Datei benötigt wird.

4.1.3.2.1 Logo von ABC-Darian (vertikal)

Als erstes Objekt in Blender wurde das Logo des Projekts entworfen. In Abbildung 4-9 sieht man die einzelnen Objekte, die es in der Szene gibt.

Abbildung 4-9: Szenenobjekte des Logos

- **Meshes**
  Die vier Würfel, die im Logo die Spielbausteine darstellen.
- **Textmeshes**
  Die Zahlen und Buchstaben, die auf den sichtbaren Seiten der Würfel zu sehen sind. Das Textmesh „txt_Darian“ ist der dreidimensionale Text, der im rechten Winkel horizontal zu den Würfeln steht (Gestaltung dreidimensionaler Texte: siehe 4.1.3.2.4).

- **Camera**
  Beschreibt den Bildschirm, der beim gerenderten Bild angezeigt wird. Mit den verschiedenen Zoom-Einstellungen kann das Fenster vergrößert werden und mit den Rotations- und Positions-Einstellungen die Richtung, in die die Linse zeigt.
- **Hemi**
  Die Lichtquelle ist über den Würfeln platziert und strahlt in einem großen Radius hinunter.

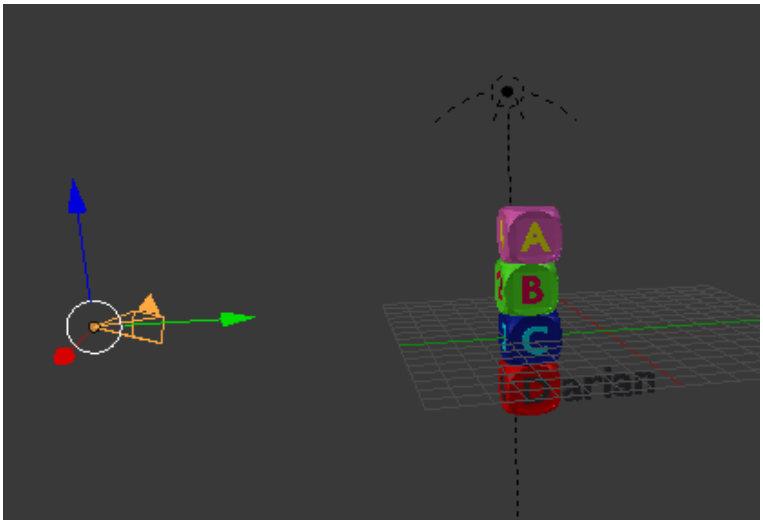


Abbildung 4-10: Logo in anderer Ansicht

Abbildung 4-10 zeigt das Logo in einer Ansicht, in der auch die Kamera und das Licht sichtbar sind.

Damit die Objekte dann vor einem schönen Hintergrund stehen, wurden die World-Einstellungen dementsprechend verändert (Farbverlauf: siehe Abbildung 4-11).

Das fertige Logo mitsamt den Farbmaterialien kann in Abbildung 4-12 betrachtet werden.

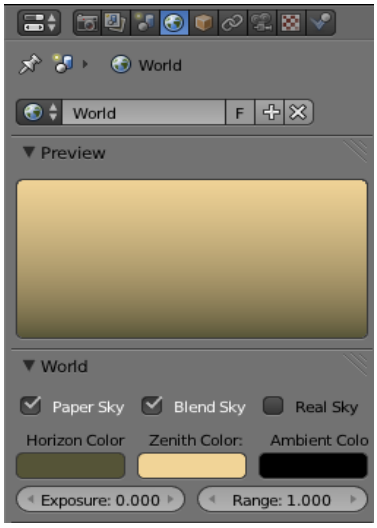


Abbildung 4-11: World-Einstellungen des Logos

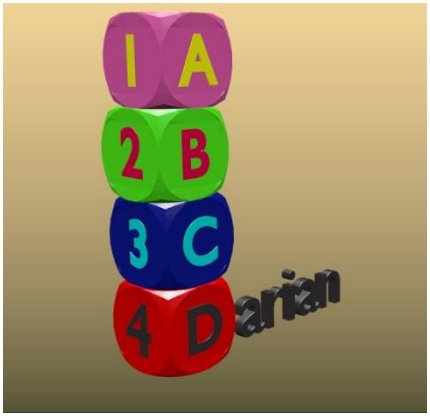


Abbildung 4-12: Logo

4.1.3.2.2 Banner von ABC-Darian (horizontal)

Er beinhaltet dieselben Objekte wie das Logo (siehe 4.1.3.2.1), sie sind aber anders angeordnet, sodass der Banner eine horizontale Skalierung besitzt, was für das Design von gewissen Dokumenten oder Präsentationen praktischer ist. Außerdem haben die Würfel und Zahlen andere Farben (Materialien wurden neu eingefärbt). Auch die World-Einstellungen wurden in den Farben geändert. Ein weiterer Unterschied besteht darin, dass der Hintergrund diesmal auch ein „Real Sky“ ist, was den in Abbildung 4-13 zu sehenden Farbverlauf ermöglicht. Das fertige Banner ist in Abbildung 4-14 abgebildet.

Abbildung 4-13: World-Einstellungen des Banners

Abbildung 4-14: Banner

4.1.3.2.3 Apfel

Um einen Apfel zu modellieren, ist ein simples Sphere-Mesh notwendig, welches im Edit-Mode editiert wird. Die Flächen müssen nur so „eingedrückt“ und gezerrt werden, dass es einem Apfel ähnelt. Der Stiel und die zwei Blätter sind wiederrum eigene, modellierte Meshes. In Abbildung 4-15 sieht man den bereits eingefärbten Apfel im Edit-Mode. Der Körper wurde markiert und man kann sehen, in welche Flächen er unterteilt ist. Auf der rechten Seite sieht man die 4 Objekte, aus denen der Apfel zusammengesetzt ist (Körper, Blätter, Stiel). Jedes davon ist mit einem Farb-Material versehen. Das Rot des Körpers hat einen hohen Specular-Wert, sodass es einen gewissen Glanzeffekt aufweist.

Der Apfel wurde für das Spiel „Apfelpyramiden“ modelliert, kommt aber auch in anderen Szenen als „Dekoration“ vor.

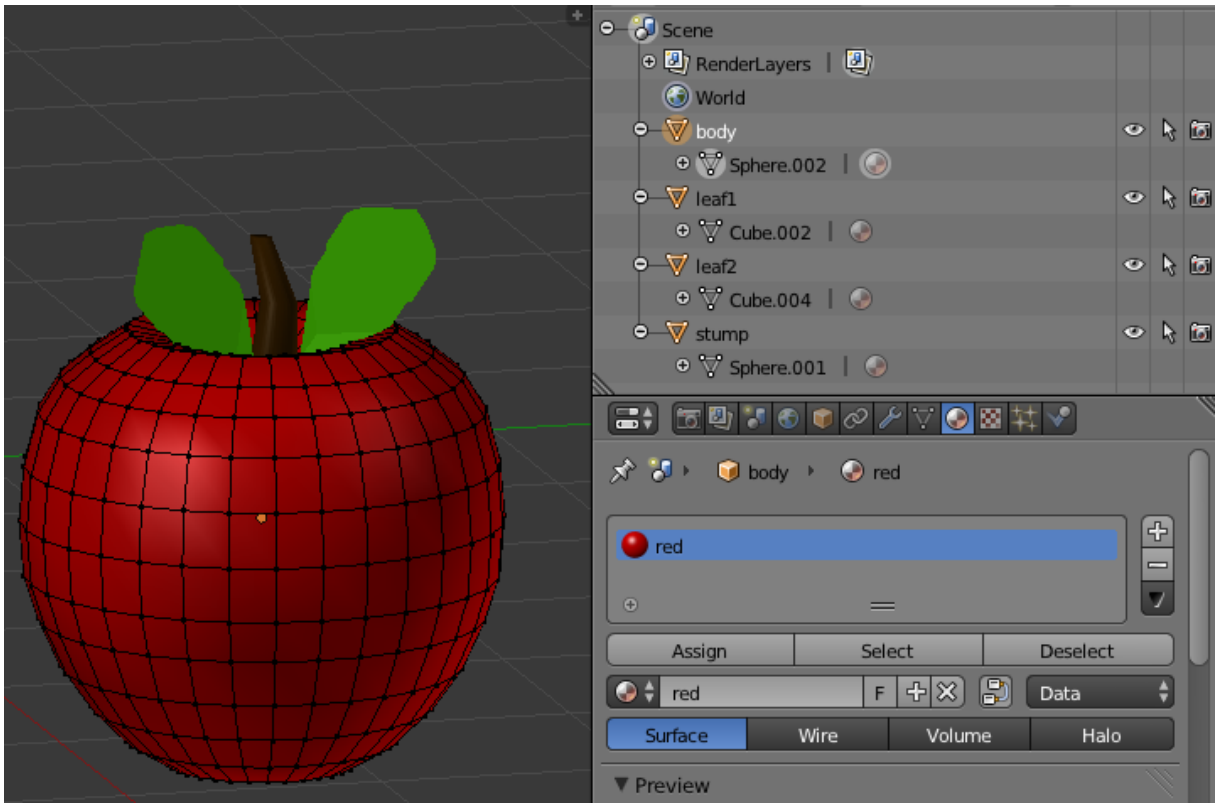


Abbildung 4-15: Apfel im Edit-Mode

4.1.3.2.4 Buchstaben und Zahlen

Dreidimensionale Buchstaben und Zahlen sind für das Arbeiten und Programmieren im dreidimensionalen Raum unbedingt notwendig. Deshalb wurden neben den Buchstaben von A-Z (inkl. Umlaute) und Zahlen von 0-9 ebenso diverse Satzzeichen modelliert.

Um einen 3D-Text in Blender zu modellieren benötigt man ein Text-Mesh. Dieses kann im Edit-Mode zu jeglicher Zeichenfolge editiert werden. In Abbildung 4-16 kann man das fertige Text-Mesh im Edit-Mode betrachtet werden, sowie seine Abmessungen. Der graue Balken rechts vom Buchstaben entspricht einem Cursor.

Damit das Text-Mesh dreidimensional wird, müssen einige Änderungen an den Einstellungen des Objekts vorgenommen werden.

In Abbildung 4-16 kann man außerdem das Geometry-Menü des Meshes sehen.

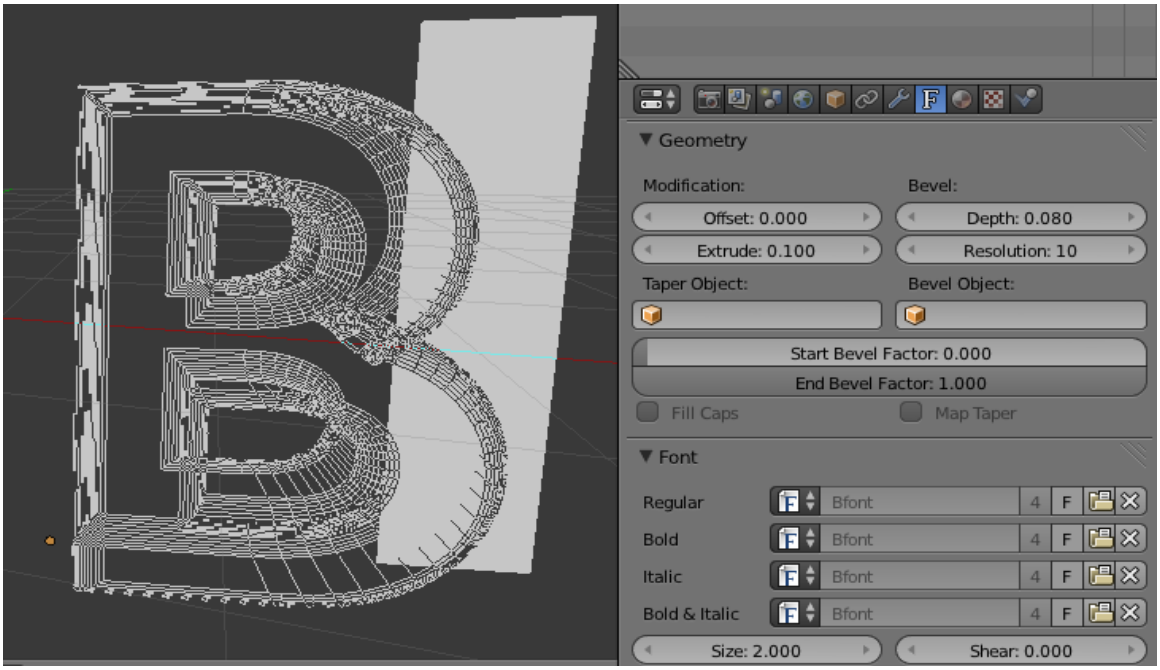


Abbildung 4-16: 3D Text-Mesh im Edit-Mode

Folgende Einstellungen bewirken das Aussehen des Buchstabens (Fertiger Buchstabe: siehe Abbildung 4-17):

- Offset „0.00“: Der Buchstabe liegt direkt auf der Zeile
- Depth „0.08“: Die Dicke des Buchstabens ist erhöht (macht ihn runder und fülliger)
- Extrude „0.10“: Die Breite des Buchstabens in Y-Richtung (macht ihn dreidimensional)
- Resolution „10.00“: Die Weiche des Buchstabens (macht ihn weniger kantig, sanfter)

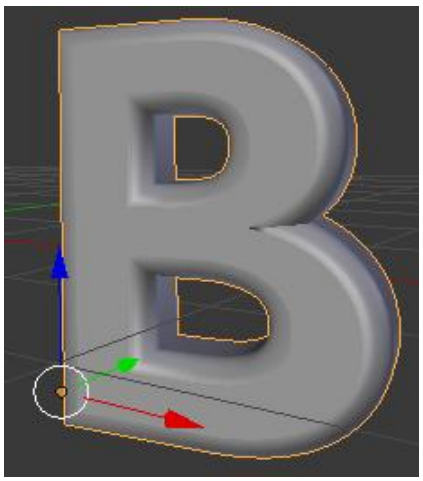


Abbildung 4-17: Buchstabe im Object-Mode

Vor dem Exportieren in ein *.cad*-File müssen die Text-Meshes mittels CTRL+C in Meshes umgewandelt werden, damit es in Unity3D keine Formatfehler gibt.

4.1.3.2.5 Buchstaben auf Hintergrund

Diese Karten mit beidseitigem Buchstabenaufdruck wurden für jeden Buchstaben des Alphabets entworfen (Groß- und Kleinschreibung). Die Modellierung erfolgte, da in manchen Legasthenie-Levels ganze Wörter aus einzelnen Buchstaben (ein Buchstabe = ein Objekt) auf andere Objekte geschrieben werden müssen. Da dies auch dynamisch möglich sein soll, sind Buchstabenfelder mit exakt gleicher Skalierungsgröße unbedingt notwendig (zwei Objekte mit der Skalierung 1x/1y/1z müssen auch den exakt gleichen Patz im dreidimensionalen Raum einnehmen, sprich gleich groß sein.) Dies wird durch die Verwendung von gleich großen Hintergründen (Cubes) ermöglicht.

Die Buchstabenkarten bestehen aus zwei Text-Meshes und einem Cube-Mesh (siehe Abbildung 4-18). Das Text-Mesh „front“ ist die um die Z-Achse gedrehte Kopie vom Text-Mesh „back“ und befindet sich auf der Hinterseite des Cubes. Der Cube an sich ist mit den Seiten Y und Z quadratisch, wobei die Seite Z nur ein 40tel der beiden anderen Längen beträgt. Somit erhält man eine bedruckte Karte, die man um die Z-Achse drehen kann und dann immer noch lesen kann.

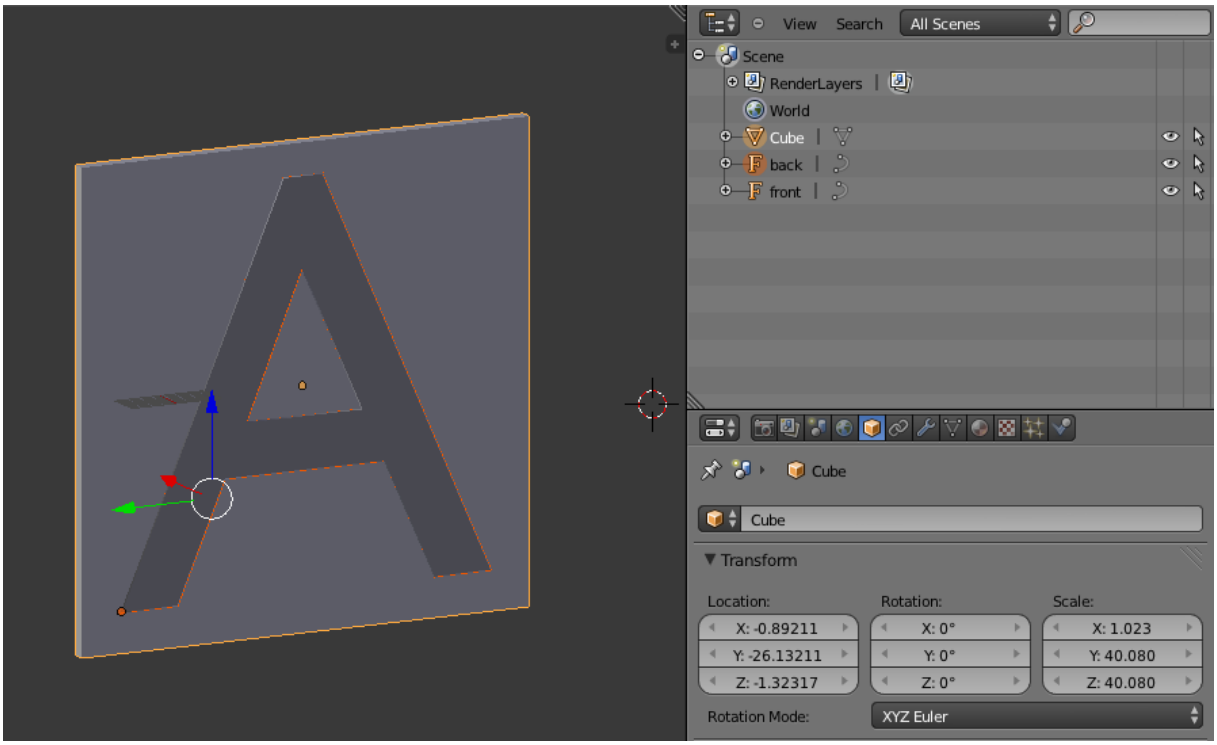


Abbildung 4-18: Buchstabenkarte im Object-Mode

Vor dem Exportieren in ein *.fbx*-File müssen die Text-Meshes mittels CTRL+C in Meshes umgewandelt werden, damit es in Unity3D damit keine Formatfehler gibt.

4.1.3.2.6 Labyrinth

Das Labyrinth wird für ein Minispiel benötigt, in dem es dem Spieler möglich sein soll, sich durch einen dreidimensionalen Irrgarten zu bewegen, auf der Suche nach Zahlen.

Das Labyrinth selbst wurde durch die Modellierung eines simplen Würfels erstellt. Die Oberseite des Würfels wurde mit dem Befehl *subdivide* in mehrere Flächen unterteilt (16 x

16). Die Flächen, welche Wände darstellen sollten, wurden nun alle gemeinsam markiert (laut einer auf Papier gezeichneten Vorlage) und auf der Z-Achse nach oben geschoben. Außerdem wurden Boden- und Wandflächen zur besseren Optik anders eingefärbt.

Das Ergebnis sieht man in Abbildung 4-19.

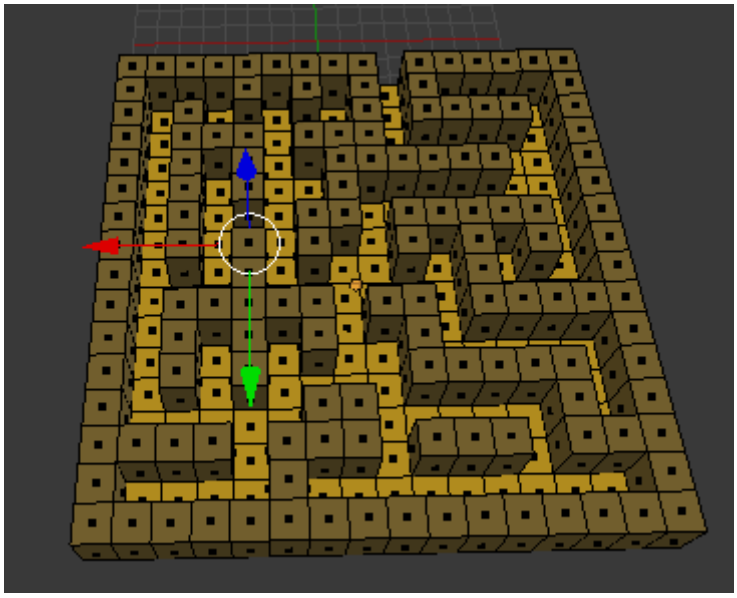


Abbildung 4-19: Labyrinth-Objekt im Edit Mode

4.1.3.2.7 Schüssel

Die Schüssel wurde für das Spiel „Essen einsammeln“ designt, da die eingesammelten Objekte von dem/der BenutzerIn zu einem Sammelplatz gebracht werden müssen.

Die modellierte, fertige Schüssel ist in Abbildung 4-21 zu sehen – die Innenflächen sind gut zu sehen –, die bereits eingefärbte in Abbildung 4-20.

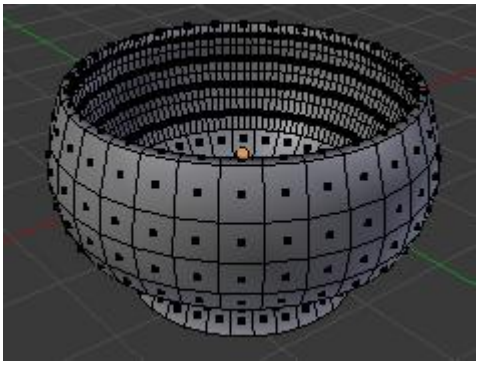


Abbildung 4-21: Schüssel im Edit-Mode

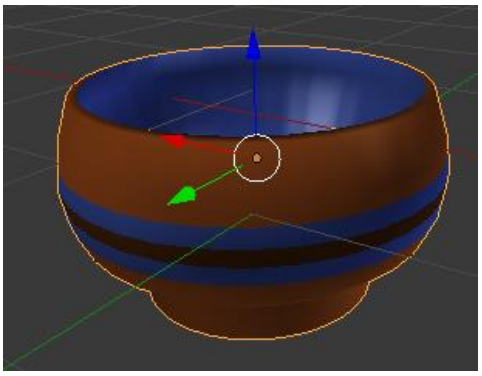


Abbildung 4-20: Fertige Schüssel

4.1.3.3 Minispiele

4.1.3.3.1 Zahlen einsammeln

Zahlen einsammeln ist ein Spiel, bei dem sich das Kind an kleine Rechenaufgaben heranwagen kann. Gerechnet wird nur im Bereich <20. Das Kind bewegt sich mit einer *First Person* durch die Welt und sammelt die Ergebnisse der gestellten Rechnungen ein.

Funktionsweise

Gesteuert wird eine *First Person* mit den Pfeiltasten. Die Ziffern sind statisch am Spielfeld platziert worden. Zu Beginn wird eine Integer-Variable zufällig mit einem Ergebnis befüllt und am oberen Spielfeldrand angezeigt.

Wenn der Benutzer in eine Ziffer hineinrennt, wird durch die Collider ein OnTriggerEnter ausgelöst und diese Ziffer wird gespeichert und angezeigt. In der Update()-Methode wird ständig abgefragt, ob das Ergebnis mit dem bereits eingesammelten übereinstimmt. Bei einstelligen Ergebnissen wird schon beim Einsammeln der ersten Zahl ausgewertet, ob das Ergebnis stimmt oder nicht. Bei mehrstelligen Ergebnissen können Fehler in der ersten Ziffer durch das Rennen in den Mülleimer, der sich ebenfalls am Spielfeld befindet, korrigiert werden. Wenn das Ergebnis richtig abgeliefert wurde, wird eine neue Rechnung gefordert. Nach 15 Runden ist das Spiel vorbei.

Assets der Szene

In Abbildung 4-22 kann man die verwendeten Assets dieser Szene sehen. Die Textfelder sind zum Anzeigen diverser Werte wie die bereits eingesammelten Zahlen, die gestellte Rechnung, ein Text, ob man gewonnen oder verloren hat, usw. Diese Szene ist die einzige, in der diese Art der Textanzeige verwendet wird (textfield.setText). Das Zahlen-Array sind die 3D-Zahlen, die im Raum verteilt sind.

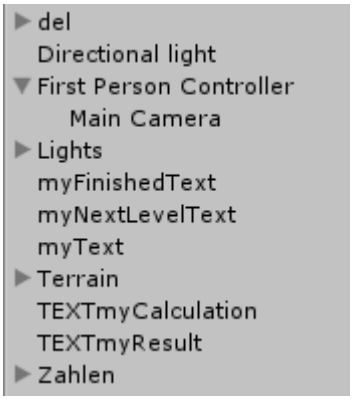


Abbildung 4-22: Assets von "Zahlen einsammeln"

4.1.3.3.2 Äpfel zählen

„Äpfel zählen“ oder „Evaluate Biggest“ ist ein Spiel, welches Kindern helfen soll, Mengen besser zu verstehen. Da Kinder Probleme damit haben, Mengen und Zahlen im Gehirn zu verknüpfen und zu verstehen, dass Mengen eine Zahl direkt darstellen können, ist dies eine gute Übung.

Algorithmus zum Erstellen der Apfelpyramide

Nachdem die zufällig gewählten Werte berechnet wurden, wird die Methode „drawApples“ aufgerufen, welche die GameObjects (Äpfel) erzeugt. Zuerst wird mittels der jeweiligen Zufallszahl (Anzahl der Äpfel) bestimmt, wie viele Äpfel sich am Boden der Pyramide befinden sollen (Variable „colMax“), und wie hoch sie sein soll (Variable „rowMax“, ergibt sich rechnerisch aus der Anzahl und dem Wert „colMax“). Danach werden sie erzeugt.

```
//---determine hight and width of pyramide
        if (max <= 28 && max >= 22)
            rowMax = 7;
        if (max <= 21 && max >= 16)
            rowMax = 6;
        if (max <= 15 && max >= 11)
            rowMax = 5;
        if (max <= 10 && max >= 7)
            rowMax = 4;
        if (max <= 6 && max >= 4)
            rowMax = 3;
        if (max <= 3 && max >= 2)
            rowMax = 2;
        colMax = rowMax; //as high as wide
```

Objekte instanziieren

Die Variable cnt ist in dem Fall eine Integer-Variable, die die bereits erstellten Äpfel mitzählt. Mit den zwei for-Schleifen ist es möglich, Reihen und Spalte zu realisieren, um einen kompakten Code für die dynamische Instanziierung der Objekte zu generieren. Jeder erstellte Apfel wird in dem GameObject Array „apples“ gespeichert. Dies diente lediglich der Kontrolle.

```
//---draw apples
for (int row = 1; row <= rowMax; row++) {
    for (int col = 1; col <= colMax; col++) {
        if (cnt < max) {
            clone = Instantiate (apple, new Vector3 ((float)((startx + row) + col * 2),
                    row * (float)1.5, 1), Quaternion.identity) as GameObject;
            cnt++;
            apples [indexForApples] = clone;
            indexForApples++;
        }
    }
    colMax -= 1; //to get less apples row by row
}
```

4.1.3.3.3 Memory

Dieses Spiel ist wie das herkömmliche Memory aufgebaut. Zu Beginn des Spiels ist eine GUI-Abfrage, mit wie vielen Karten der Benutzer spielen möchte (8, 16, 24 oder 36). Danach spielt man wie gehabt, Eingabe ist ein Mausklick.

Skripts

Memory ist ein sehr aufwendiges Spiel, welches in Summe drei Skripts für den Ablauf verwendet. In Abbildung 4-23 werden sie und ihre zugewiesenen Variablen gezeigt (alle Skripts sind Komponenten der Kamera).

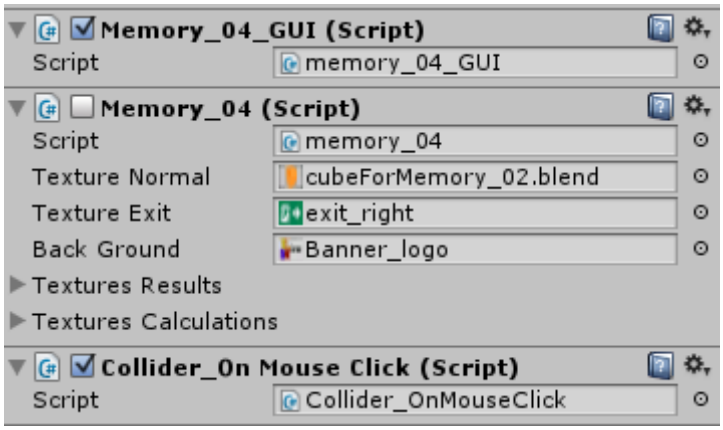


Abbildung 4-23: Skripts in Memory

Das Skript „memory_04“ ist jenes, welches den generellen Spieleablauf beinhält. Es ist zu Beginn ausgeschaltet, da zuerst nur die GUI angezeigt werden soll, in welcher der Benutzer die Anzahl der Karten bestimmen kann. Das Skript „memory_04_GUI“ macht dies möglich. Es ist schon zu Anfangs aktiviert.

Das Skript „Collider_OnMouseClick“ beinhält die Erlaubnis und Erkennung des Klicks auf die Karten. Wird eine Karte geklickt, so wird ein Trigger ausgelöst welcher es ermöglicht, auf das Ereignis zu reagieren (setzen diverser Variablen, damit auch in „memory_04“ das Ereignis bekannt ist).

Texturen

Für diesen Level mussten sehr viele Texturen in Paint gezeichnet werden, denn jede Rechnung und jedes Ergebnis werden mittels Textur auf die Karten gespielt. Es gibt insgesamt 20 verschiedene Ergebnisse, auf die jeweils wiederrum drei passende Rechnungen kommen. Dies gewährleistet viel Abwechslung.

In Abbildung 4-24 und Abbildung 4-25 sind Beispiele dieser Texturen zu sehen.

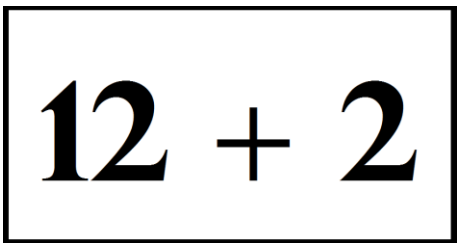


Abbildung 4-24: Rechnungstextur

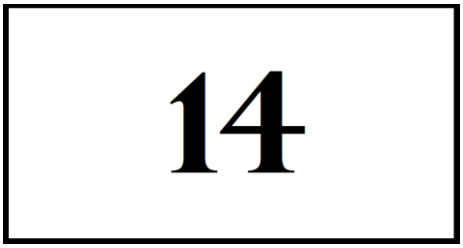


Abbildung 4-25: Ergebnistextur

Funktionsweise

Nachdem der Nutzer in der anfänglichen Auswahl die Anzahl der Karten bestimmt hat, werden dementsprechend viele generiert. Das Generieren ist ein aufwendiger Prozess, der in folgende Teilaufgaben gegliedert ist:

- Zufällige Auswahl der Ergebnistexturen (gespeichert in Array)
- Zufällige Auswahl der passenden Rechnungstexturen (gespeichert in Array)
- Durchmischen der erhaltenen Texturen, um eine dynamische und rein zufällige Anordnung der Spielkarten zu erhalten
- Erstellen von flachen Würfeln mit selben Abmessungen und Abständen
- Speicherung von Überprüfungsdaten in mehreren Boolean-Arrays (damit später noch nachvollzogen werden kann, welche beiden Würfel ein Pärchen ergeben)
- Versehen der Würfel mit den Texturen

Da in diesem Spiel vieles gleichzeitig geschehen sollte und daher mit Echtzeit gearbeitet wird, wurden einige Threads erzeugt. Dies ist mit dem Methodenrückgabewert *IEnumerator* und dem Befehl *waitForSeconds(float time)* möglich. Für folgende Algorithmen wurde ein Thread geschrieben:

- Umdrehen der Karten sowie Änderung ihrer Textur in diesem Vorgang
- Auswertung zweier geklickter Karten (Pärchen oder nicht?)
- Löschen der Spielekarten von gefundenen Pärchen

Auswahl der Kartenanzahl

Hier sieht man die Anzeige der Kartenauswahl. Auf die Variable chosenAmount wird dann im Skript „memory_04“ mittels Getter zugegriffen.

```
GUILayout.BeginArea (new Rect (w / 3, h / 4 + h/8, w / 3, h / 2));
if (GUILayout.Button ("8", buttonStyle)) {
      chosenAmount = 8;
}
if (GUILayout.Button ("16", buttonStyle)) {
      chosenAmount = 16;
}
if (GUILayout.Button ("24", buttonStyle)) {
      chosenAmount = 24;
}
if (GUILayout.Button ("36", buttonStyle)) {
      chosenAmount = 36;
}
GUILayout.EndArea ();
```

Algorithmus zur Zufallszuweisung der Texturen

Mit dem Knuth-Algorithmus ist es möglich, den Inhalt eines Arrays zufällig durchzumischen. Dies dient dazu, dass die im Vorfeld dynamisch ausgewählten Texturen (gespeichert in dem Textures Array „chosenTextures“) nicht nach der Reihe am Feld angeordnet werden. Somit wird die Schwierigkeit vergrößert. Dieser Vorgang ähnelt dem Durchmischen der Spielkarten bei der Brettspielausgabe des Spiels.

```
// Knuth shuffle algorithm : shuffle array
for (int t = 0; t < chosenTextures.Length; t++) {
      Texture tmp = chosenTextures [t];
      int r = Random.Range (t, chosenTextures.Length);
      chosenTextures [t] = chosenTextures [r];
      chosenTextures [r] = tmp;
}
```

4.1.3.3.4 Würfel zählen

Im Spiel Würfel zählen wird ein Turm aus Blöcken (Würfeln) erzeugt. Die Anzahl der Blöcke wird zu Beginn des Spiels zufällig generiert. Die Aufgabe des Kindes ist es nun, um die Blöcke herumzulaufen und sie abzuzählen.

Funktionsweise

Die zufällig generierte Anzahl der Würfel wird zu Beginn ausgewertet und dann durch die Anzahl der Würfel, die in einer Reihe liegen, dividiert. Somit weiß man, wie hoch der Turm sein wird. Der Rest, der bei der Division entsteht, wird in eine Variable gespeichert.

Nun werden alle Würfel der Reihe nach mit einem definierten Abstand dazwischen instanziiert. Für die letzte Reihe, in der sich wegen der Division meist weniger Würfel befinden, als in den anderen Reihen, muss eine eigene Abfrage generiert werden.

Gibt der Benutzer nun sein Ergebnis ein, wird dieses beim Drücken der Enter-Taste ausgewertet.

Benutzereingabe

Die Benutzereingabe erfolgt über die Tastatur. Hierfür muss in der Update()-Methode ständig auf Tastendrucks abgefragt werden. Wenn einer erkannt wurde, so wird die jeweilige Ziffer dem Ergebnis-String angefügt.

```
                if (Input.GetKeyDown (KeyCode.Alpha0) || Input.GetKeyDown
(KeyCode.Keypad0))
                    strResult += "0";
                else if (Input.GetKeyDown (KeyCode.Alpha1) || Input.GetKeyDown
(KeyCode.Keypad1))
                    strResult += "1";
```

4.1.3.3.5 Wrong Or Right

In diesem Spiel muss man erkennen, ob die angezeigte Rechnung korrekt ist oder nicht. Hierzu kann mit der Kinect oder der Tastatur gesteuert werden.

Funktionsweise

Es werden zufällig Rechnungen in einem definierten Zahlenbereich erstellt. Diese müssen auf mehrere Arten gespeichert werden:

- als String, der das korrekte Ergebnis beinhält
- als String, der das anzuzeigende Ergebnis beinhält
- jede Teilnummer als Integer, um zufällig Fehler einzubauen

Es wird mit einer boolschen Zufallsvariable entschieden, ob die aktuelle Rechnung einen Fehler haben soll, oder nicht. True bedeutet Fehler, False bedeutet keiner. Wenn ein Fehler eingebaut werden soll, wird ein zufälliger Integer-Wert im Wertebereich von 1-3 erzeugt, der dann vom echten Ergebnis abgezogen oder dazu addiert wird. Standartmäßig wird dieser „Error“ vom Ergebnis abgezogen, doch per Zufall wird wieder entschieden, ob der doch dazu addiert werden sollte.

```
        //default: substract error from real result and set result
                int error = Random.Range (0, 3);
                res -= error;
```

```
//for alternation: randomly add the error (or if the result is negativ)
if (Random.Range (0, 2) > 0)
    res += 2 * error;
```

Sobald der Benutzer dann auf eine der Steuermöglichkeiten eine Eingabe getätigt hat, wird die Antwort ausgewertet und entsprechendes angezeigt. Bei korrekter Antwort erhöhen sich die Punkte, bei falscher wird ein Punkt abgezogen. Das Spiel ist außerdem in Schwierigkeitsstufen unterteilt. Je mehr Punkte man hat, desto höher ist der eigene Level und desto Schwieriger sind die Operanden, die in den Rechnungen vorkommen. Auch der Zahlenbereich erhöht sich.

Anzeige, ob man richtig liegt

Nachdem man seine Entscheidung getroffen hat, wird man informiert, ob man richtig oder falsch lag. Dies wird in der OnGUI() erledigt.

```
if (WrongOrRight_ClickDetection.getSolved ()) { //do only if user solved
    if (WrongOrRight_ClickDetection.getUserInput ()) {
        reactToUserInput ((("Richtig,\ndenn\n" + correctText), Color.green, true,
textureGood);
    } else {
        reactToUserInput ((("Leider\nFalsch!\n" + correctText), Color.red, false,
textureBad);
    }
}
```

4.1.3.3.6 Labyrinth

Im Spiel Labyrinth ist es dem Benutzer möglich, sich durch ein dreidimensionales Labyrinth fortzubewegen. Für bessere Übersicht wurde hierfür eine Minimap an den oberen rechten Bildschirmrand eingefügt, welche das Labyrinth in der Vogelperspektive zeigt.

Der Spieler muss einen vorgegebenen Zahlenwert durch Sammeln der einzelnen Summanden im Labyrinth erreichen. Die Additionsfähigkeiten des Kindes werden dadurch trainiert.

Zahlen erstellen

Zu Beginn wird ein zufälliges Ergebnis errechnet. Dieses wird nun wiederrum durch Anwendung von Zufallsgeneratoren in fünf Summanden unterteilt, welche alle in ein Array gespeichert werden. Um die Schwierigkeit zu erhöhen, werden zusätzlich zu diesen „richtigen“ Ziffern noch „falsche“ generiert, wieder zufällig, und abgespeichert. Die Koordinaten, an welchen Ziffern gespawnt werden sollen, sind in einem Array statisch eingetragen. Für jede Zahl wird nun zufällig bei ihrer Erstellung ein Koordinatenset mit ausgewählt. Somit ist gewährleistet, dass die Zahlen sich nur in den Ecken befinden.

Nachdem der Prozess mit der Auswahl und Berechnung der Zahlen und ihrer Koordinaten fertig ist, müssen alle Zahlen instanziiert werden. Es ist wichtig, ihnen bestimmte Komponenten zu geben. Im Array numbers[] sind die Gameobjekte gespeichert (die dreidimensionalen Zahlen), im Array vecs[] die Koordinaten des Objekts in Form vom Datentyp Vector3.

```
curObj = Instantiate (numbers [index], vecs [bool_index], Quaternion.identity) as
GameObject;
```

```
curObj.transform.localScale = new Vector3 (0.5f, 0.5f, 0.5f);
curObj.AddComponent<RotateObjectsSlowly> ();
curObj.AddComponent<MeshCollider> ().isTrigger = true;
```

- **RotateObjectsSlowly**: Ist ein Skript, in dem in gewissen Zeitabständen der Rotationswert der Y-Achse des jeweiligen Objekts (this.GameObject) manipuliert wird. Somit Drehen sich die Zahlen.
- **MeshCollider**: Dieser muss getriggert werden, damit ein Zusammenstoß mit der Third Person erkannt und darauf reagiert werden kann.

Spiel beenden

In der Update()-Methode steht folgender Code:

```
if (curUserRes == result) {
    GameObject.Destroy (GameObject.Find ("door"));
    winTxt = "Richtig! Die Tür ist offen.";
}
```

Wenn die Variable curUserRes (beinhält den zurzeit vom User eingesammelten Wert) den gleichen Integer-Wert enthält, wie die Variable result (beinhaltet das zu erreichende Ergebnis), so hat man gewonnen. Es wird das GameObject „door“ zerstört, welches einen Teil der Wand des Labyrinths darstellt. Somit ist es nun möglich, nach draußen zu gehen und das Spiel somit zu beenden.

Minimap

Die Minimap ist ein Kameraobjekt, welches aber nicht über den ganzen Bildschirm angezeigt werden soll. Mit den Einstellungen des Viewport Rect kann man dies anpassen.

- X: Startpunkt in X-Richtung
- Y: Startpunkt in Y-Richtung
- W: Breite in Prozent die die Kamera vom gesamten Bildschirm einnehmen soll
- H: Höhe in Prozent die die Kamera vom gesamten Bildschirm einnehmen soll

In Abbildung 4-26 sieht man den Vergleich der Einstellungen der normalen Kamera und der Minimap.

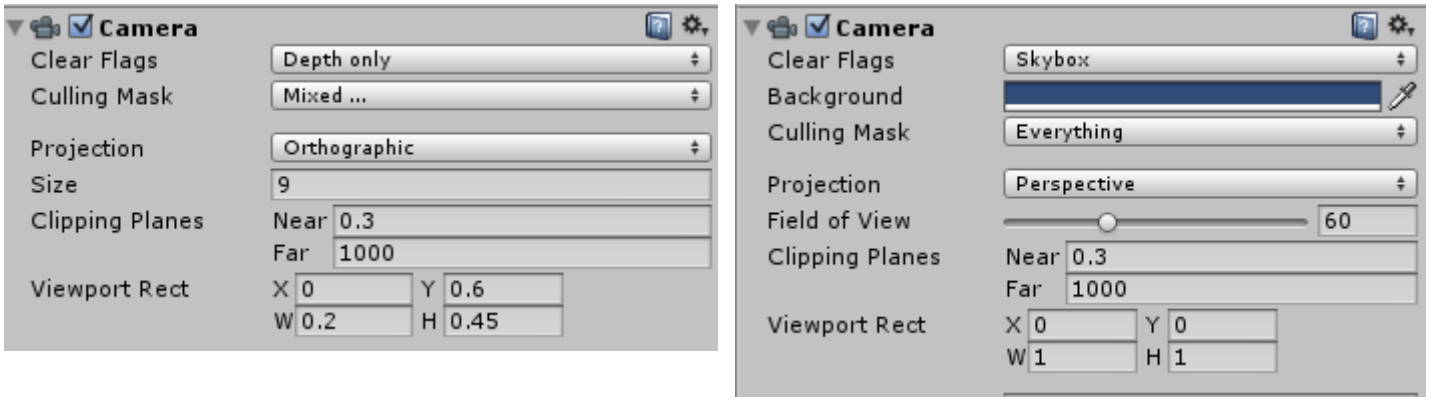


Abbildung 4-26: Vergleich der Kameraeinstellungen (links: Minimap)

4.1.3.3.7 Futter einsammeln

In diesem Spiel steuert man einen Hasen durch eine Ebene. Er muss Futter einsammeln, welches in Form von rotierenden Objekten am Rasen verteilt ist. Das Futter muss immer wieder in eine Schüssel gelegt werden.

Funktionsweise

Zu Beginn werden am Terrain an zufälligen Koordinatenpunkte 30 Futter-Objekte erzeugt (diese Objekte wurden im Asset Store runtergeladen und sind im Array foods[] gespeichert). Danach wird eine zufällige Integer-Variable erzeugt, die angibt, wie viele Futterstücke der Spieler einsammeln muss. Die Methode spawnFood(int amount) instanziiert die Futter-Objekte. Jedes erstellte Objekt wird im Array spawnedFood[] gespeichert, damit es später wieder gelöscht werden kann.

```
private void spawnFood (int amount)
{
    //define which food should be spawned first
    int foodCnt = Random.Range (0, foods.Length); //current index of food-array
    float coord_x, coord_z, coord_y = 0.3f;
    for (int i = 0; i < amount; i++) {
        //getting random coordinates in given area
        coord_x = Random.Range (-30f, 31f);
        coord_z = Random.Range (-40f, 31f);
        //spawn it
        GameObject curObj = Instantiate (foods [foodCnt], new Vector3
(coord_x, coord_y, coord_z), Quaternion.identity) as GameObject;
        curObj.transform.localScale = new Vector3 (2f, 2f, 2f);
        curObj.AddComponent<RotateObjectsSlowly> ();
        curObj.AddComponent<MeshCollider> ().isTrigger = true;
        spawnedFood [i] = curObj;
        //change food to create
        foodCnt++;
        if (foodCnt > 3)
            foodCnt = 0;
    }
}
```

Da der Hase nur vier Stücke gleichzeitig tragen kann, wird jedes Mal, wenn das Kind noch ein Objekt aufsammeln möchte, ein Warnungs-Text angezeigt.

Schüssel befüllen

Jedes Mal, wenn der Hase Futter zur Schüssel bringt, soll die Schüssel auch um genausoviele Objekte voller werden. Hierfür wurde die Methode fillBowl() geschrieben.

In dieser werden zufällige Futter-Objekte in der Schüssel erzeugt. Auch diese Objekte werden wieder in ein Array gespeichert, um sie bei Neustart der Runde zerstören zu können.

```
private void fillBowl (int amount)
{
    //fill bowl with food when filled
    for (int i = 0; i < amount; i++) {
        float coord_x = Random.Range (5.7f, 6.3f);
        float coord_y = Random.Range (0.06f, 0.4f);
        float coord_z = Random.Range (0.08f, 0.6f);

        GameObject food = Instantiate (foods [Random.Range (0, amount + 1)], new Vector3
(coord_x, coord_y, coord_z), Quaternion.identity) as GameObject;
```

```
            foodInBowl [foodArrayCnt] = food;
            foodArrayCnt++;
        }
    }
```

Neue Runde starten

Wenn das Kind eine Runde beendet hat (in dem es entweder richtig viele Objekte in die Schüssel legte oder zu viel), wird eine neue mit der Methode resetRound() gestartet. In dieser werden alle Futter-Objekte zerstört.

```
private void resetRound ()
{
    //destroy old objects
    for (int i = 0; i < spawnedFood.Length; i++) {
        Destroy (spawnedFood [i]);
    }
    //destroy bowl food
    for (int i = 0; i < foodInBowl.Length; i++) {
        if (foodInBowl [i] != null) {
            Destroy (foodInBowl [i]);
        }
    }
    startNew ();
}
```

In der Methode startNew() werden verschiedene Zähler resettet und die Methode spawnFood() aufgerufen.

4.1.3.4 Highscore-Liste

Die Highscore-Liste benötigt die Verwendung von persistenten Daten. Hierfür wurde die Klasse PlayerPrefs verwendet.

Es gibt für jedes Minispiel ein eigenes Skript, welches die Highscore-Daten dieses Spiels speichert. Dies ist nötig, damit sich die Daten nicht vermischen.

Die Highscore-Liste selbst wird dann in einer eigenen Szene aufgerufen.

4.1.3.4.1 Daten in die Highscore-Liste speichern

Mit diesem Befehl werden im Spiel „Wrong Or Right“ (Beispiel) der Name des aktuellen Spielers und dessen erreichte Punkte in die Highscore-Liste gespeichert.

```
HSM_WrongOrRight._instance.SaveHighScore (SettingsGUI.getPlayerName, points);
```

Im Skript der Highscore-Liste (HSM_WrongOrRight) sieht diese Datenverarbeitung dann folgendermaßen aus:

Der Name als String, als auch der Punktewert als Integer werden in einer for-Schleife in die Highscore-Liste geschrieben.

```
PlayerPrefs.SetString ("HighScore" + i + "name", highScore [i - 1].name);
PlayerPrefs.SetInt ("HighScore" + i + "score", highScore [i - 1].score);
```

4.1.3.4.2 Szenendesign

In der Szene befindet sich nur die Main Camera. Auf diese ist ein Skript gelegt, welches die richtige Highscore-Liste aktiviert. Dies ist nur möglich, wenn man der Szene nicht nur den Spielernamen und die Punkte übergibt, sondern auch einen Identifier. Dieser ist in dem Fall einfach eine Integer-Variable, die eine switch-Anweisung steuert. Jede Highscore-Liste wird einem Integer-Wert zugeordnet. Somit kann in diesem Skript ausgewählt werden, in welche der Highscore-Listen die Daten des Spielers gespeichert werden.

```
switch (ScriptToUse) { //give each script a number
        case 1:  //CollectNuts
        HSM_CollectNuts._instance.SaveHighScore (newName, newPoints); //save result
        break;
```

Es ist nicht nötig, die Skripts der Highscore-Listen einem Objekt zuzuordnen, weil sie automatisch auf die OnGUI zugreifen, wenn das Skript verwendet (also etwas in ihm gespeichert) wird.

4.1.4 Ergebnisse

4.1.4.1 Minispiele/Szenen

4.1.4.1.1 Highscore-Liste

In Abbildung 4-27 sieht man ein Beispiel einer Highscore-Liste.

Abbildung 4-27: Highscore-Liste vom Spiel "Futter sammeln"

4.1.4.1.2 Startmenü

Im Startmenü kann folgendes ausgewählt werden (siehe Abbildung 4-28):

Abbildung 4-28: Startmenü

Namen eingeben

Wird auf „Namen eingeben“ gedrückt, kann der Name des Spielers eingegeben werden (siehe Abbildung 4-29). Ist dieser Name bereits vorhanden, so wird dies erkannt.

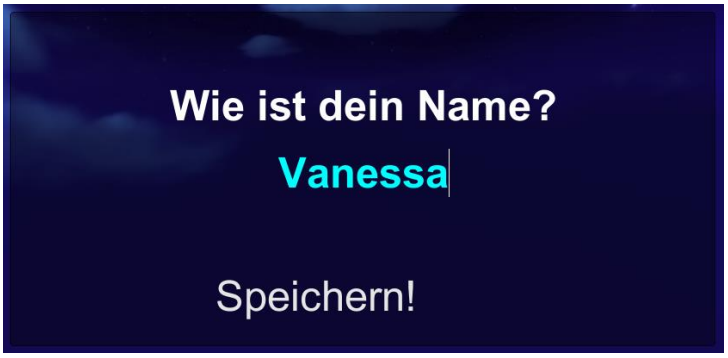


Abbildung 4-29: Namen eingeben

Laden

Bei Klick auf „Laden“ erscheinen neue Buttons, welche es dem Benutzer ermöglichen, sich zwischen Legasthenie- und Dyskalkulielevel zu entscheiden. Je nach Auswahl wird man auf die jeweilige nächste Szene (Levelauswahl) umgeleitet.

Einstellungen

Auswahl der Audioeinstellungen.

Beenden

Durch Klick auf „Beenden“ kommt eine andere GUI zum Vorschein. Hier kann abgebrochen oder das Beenden des Programms durchgeführt werden (siehe Abbildung 4-30). Bei „Ja, Beenden“ wird das Programm geschlossen und bei „Nein, Abbrechen“ kommt man zurück zum ursprünglichen Startmenü.

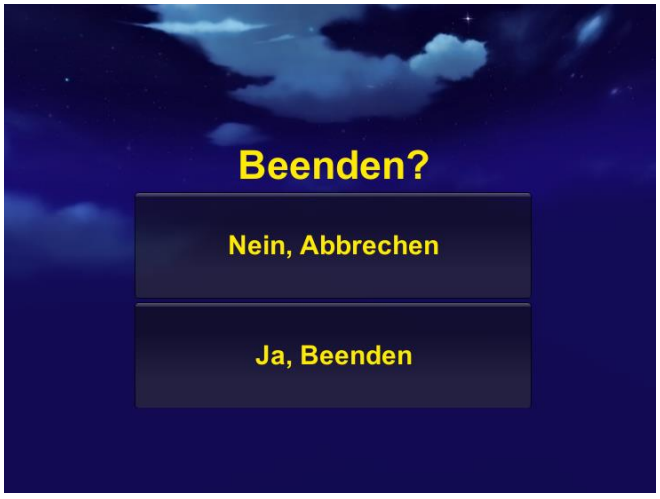


Abbildung 4-30: Spiel beenden

Um die individuelle GUI zu erstellen, wird die Methode OnGUI() verwendet, welche in Unity3D zahlreiche Features beinhält.

Mittels Änderung des Kamera-Zooms ist ein Effekt zu Beginn dieser Szene möglich.

4.1.4.1.3 Instruktionsbildschirm

Am Instruktionsbildschirm (siehe Abbildung 4-31) kann man vor jedem eigentlichen Minispiel sehen, was zu erledigen ist. Da es dem Kind Schwierigkeiten bereiten könnte, die Anweisungen alleine zu lesen, wird außerdem ein Audio-File abgespielt, welches alles vorliest. Klickt man auf den Button mit dem Lautsprecher-Symbol in der rechten unteren Ecke, so wird das Audio-File erneut abgespielt. Bei Betätigung des Buttons unten in der Mitte wird dann das Minispiel geladen.

Abbildung 4-31: Instruktionsbildschirm (Bsp: Memory)

4.1.4.1.4 Apfelpyramiden

Die Spieleansicht sieht aus wie in Abbildung 4-32 dargestellt.

Das Kind muss sich entscheiden, welche Menge (bzw. welche Zahl) größer (höher) ist. Hierzu werden zwei Integer Variablen (ganze Zahlen von 1 bis 28) zufällig generiert. Auf der linken Bildschirmhälfte wird der erste Wert dargestellt, auf der rechten Bildschirmhälfte der zweite. Die Darstellung erfolgt einerseits in Form einer Pyramide aus Äpfeln, andererseits aus der Zahl selbst, die über den Äpfeln angezeigt wird. Ein Apfel repräsentiert dabei die Einheit „1“, sodass es die Aufgabe des Kindes ist, die größere Menge entweder durch Abzählen oder durch Abschätzung zu bestimmen. Der bereits erwähnte Zahlenwert über der Pyramide wird aber nur abhängig von der Schwierigkeitsstufe angezeigt („leicht“ ist mit Zahlenwert, „schwierig“ ist ohne).

Wenn sich das Kind entschieden hat, muss es einfach den Button über der jeweiligen Apfelpyramide drücken. Danach wird angezeigt, ob man richtig oder falsch gelegen hat, der Punktezähler und Versuchszähler ändern sich dementsprechend, und neue Pyramiden bzw. Werte werden generiert.

Sind die Versuche aufgebraucht (Lebensanzeige in der oberen linken Ecke), so wird der Spielstand gespeichert und in die Bestenliste eingetragen. Selbiges passiert, wenn das Spiel manuell mittels Exit-Button (untere rechte Ecke) beendet wird.

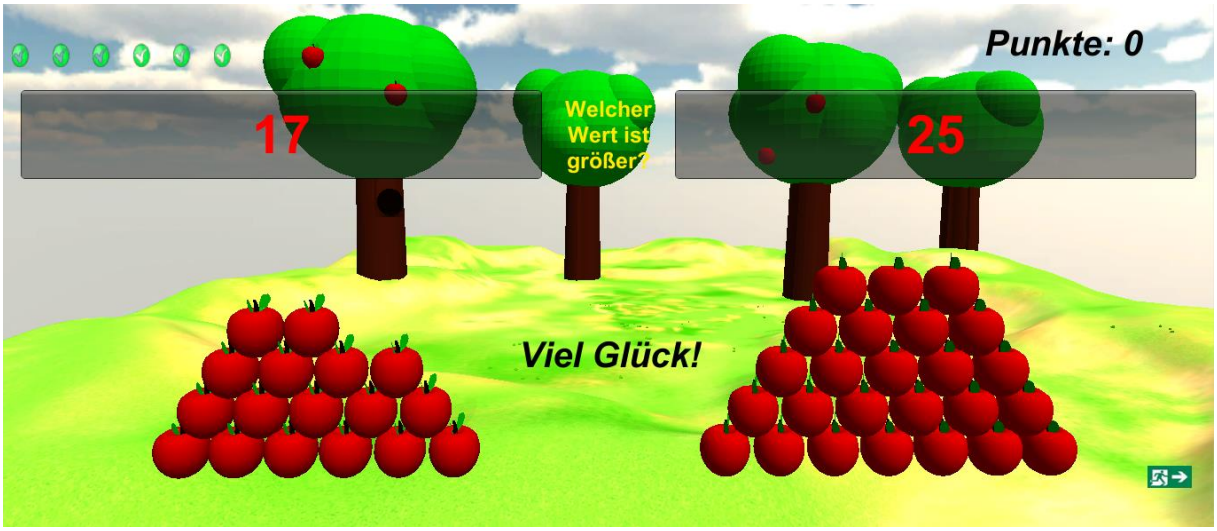


Abbildung 4-32: Spieleansicht von „Äpfel zählen“

4.1.4.1.5 Würfel zählen

Die Spieleansicht sieht aus wie in Abbildung 4-33 dargestellt.

Der Benutzer kann mit den Pfeiltasten und der Maus um die Blöcke herumgehen. Ziel ist es, sie richtig abzuzählen. Dies gibt Kindern ein Verständnis für Mengen und ihre Repräsentation. Das Ergebnis kann mit der Tastatur eingegeben werden. Bei Betätigung von Enter wird das Ergebnis ausgewertet. Liegt man richtig, so bekommt man eine neue Herausforderung und der Punktezähler erhöht sich. Liegt man falsch, verliert man einen Punkt und versucht es erneut. Gespielt wird solange, bis manuell beendet wird.

Abbildung 4-33: Spieleansicht "Würfel zählen"

4.1.4.1.6 Zahlen einsammeln

Die Spieleansicht sieht aus wie in Abbildung 4-34 dargestellt.

Das Kind spielt in der First Person Ansicht. Oben rechts wird die Rechnung angezeigt, die gelöst werden muss. Das Ergebnis lautet in dem Fall 2, also muss das Kind einfach die 2 am Spielfeld suchen und in sie hineinlaufen. Stimmt es, so geht es mit der nächsten Rechnung weiter. Wenn nicht, muss man es nochmal versuchen. Zur Fehlerbehebung von zweistelligen Zahlen gibt es den Mülleimer.

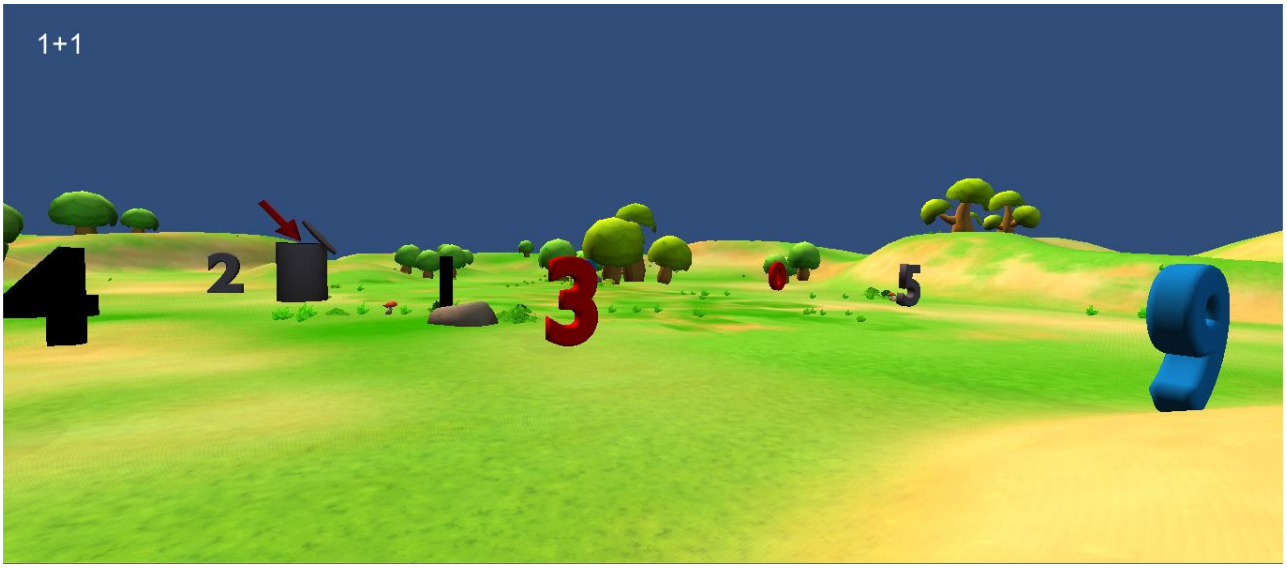


Abbildung 4-34: Spieleansicht von „Zahlen einsammeln“

4.1.4.1.7 Memory

Die Spieleansicht ist wie in Abbildung 4-35 dargestellt.

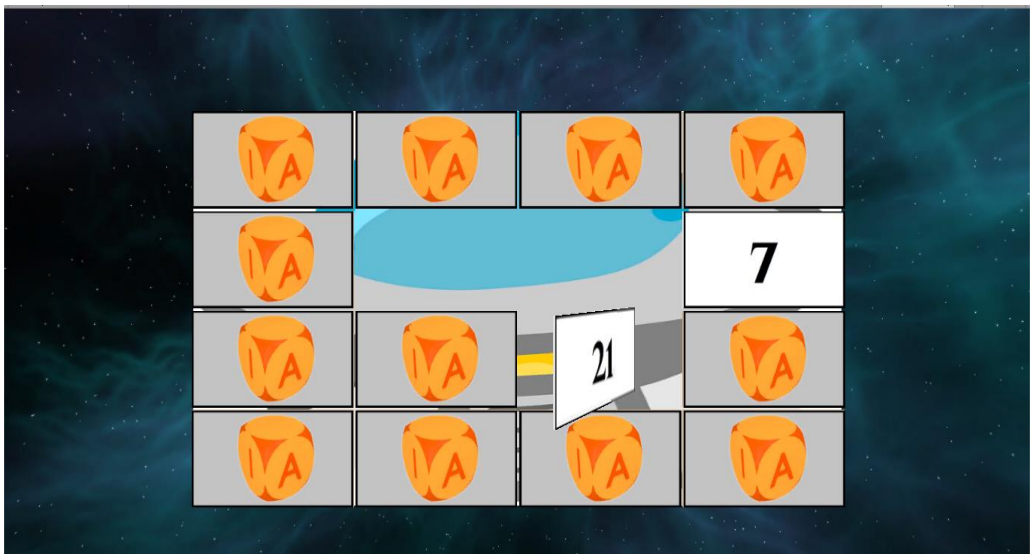


Abbildung 4-35: Spieleansicht des Spiels „Memory“

Zuerst kann der Benutzer auswählen, mit wie vielen Karten er spielen möchte (siehe Abbildung 4-36). Danach kann auch schon losgelegt werden.

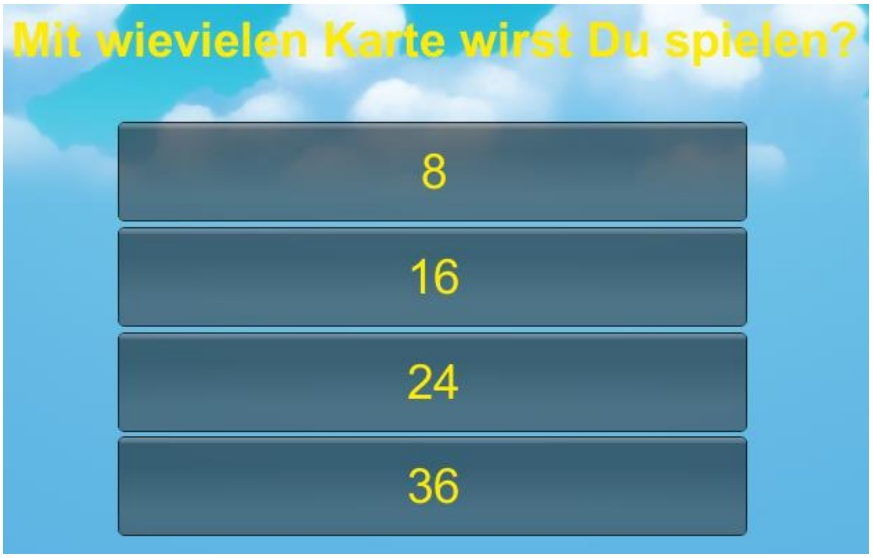


Abbildung 4-36: Kartenanzahl auswählen

Ein Pärchen bildet jeweils eine kurze Rechnung und ihr dazu passendes Ergebnis. Es können immer nur zwei Karten zugleich aufgedeckt werden. Wenn man zwei passende Karten gefunden hat, werden diese gelöscht und man kann weiter suchen und rechnen. Wenn das Spiel fertig ist, also alle Kärtchen vom Tisch sind, wird man gefragt, ob man noch einmal spielen möchte.

4.1.4.1.8 Wrong Or Right

Dieses Spiel ist mit der Kinect spielbar. Die Ansicht des Spielers kann man in Abbildung 4-37 betrachten.

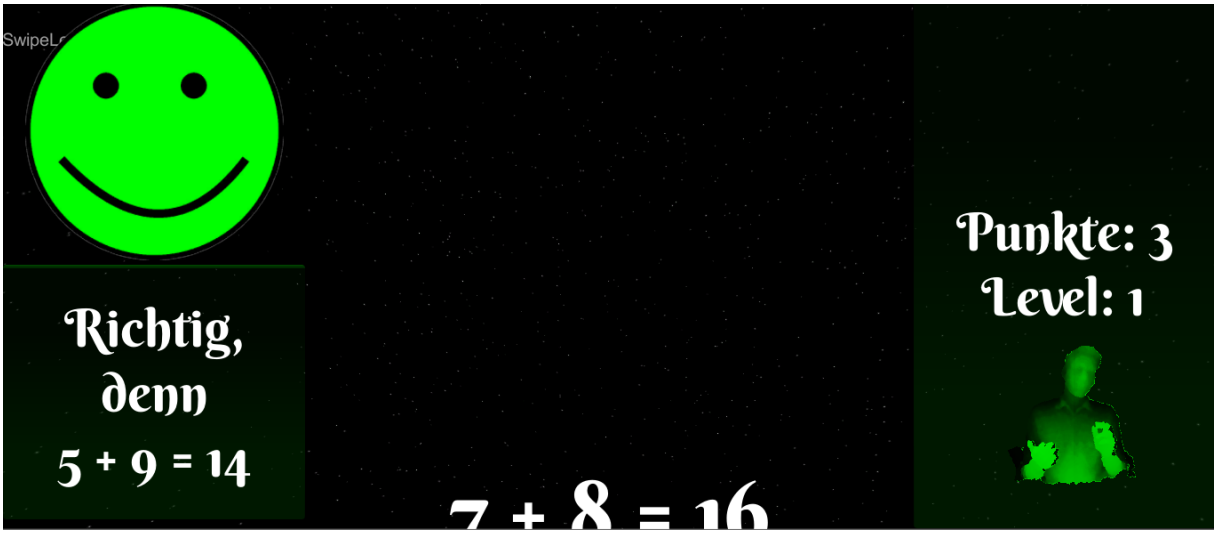


Abbildung 4-37: Spieleansicht "Wrong Or Right"

Der Spieler muss herausfinden, ob die Rechnung, die ihm gezeigt wird (sie kommt von unten und wandert den Bildschirm nach oben) stimmt.

- Nach links wischen: Die Rechnung ist richtig
- Nach rechts wischen: Die Rechnung ist falsch

Ob man mit seiner Annahme richtig gelegen hat, wird oben links am Bildschirm angezeigt. Unten links wird außerdem die Rechnung aufgelöst.

Seine Punkte und das momentane Level sieht man auf der rechten Seite. Darunter kann der Spieler sehen, dass er von der Kinect erkannt wird.

4.1.4.1.9 Futter einsammeln

Die Spieleansicht kann in Abbildung 4-38 betrachtet werden.

In diesem Spiel lernen Kinder Addieren und Zählen. Die Aufgabenstellung ist es, den Hasen Futter sammeln zu lassen und es in eine Schüssel zu legen. Oben rechts am Bildschirm steht, wieviel er haben möchte, darunter, wie viel er gerade trägt, und darunter, wieviel gerade in der Schüssel liegt. Der Hase kann nur vier Stücke auf einmal tragen, man muss also immer wieder zur Schüssel gehen und sein Futter dort lagern. Wenn man erkennt, dass man aus Versehen zu viel Futter trägt, so kann man ein Stück mit „F“ fallen lassen.

Legt man zu viel in die Schüssel, so verliert man ein Leben (Anzeige oben links). Legt man aber genau richtig viele Futterstücke in die Schüssel, so gewinnt man Punkte und kommt in die nächste Runde. Das Spiel ist vorbei, sobald man keine Leben mehr übrig hat.

Abbildung 4-38: Spieleansicht "Futter einsammeln"

4.1.4.1.10 Labyrinth

Die Spieleansicht kann in Abbildung 4-39 betrachtet werden.

Abbildung 4-39: Spieleansicht "Labyrinth"

In diesem Spiel muss der/die BenutzerIn in einem Spiel die Summanden eines vorgeschriebenen Ergebnisses finden. Doch es sind nicht nur die richtigen Ziffern im Irrgarten zu finden, sondern auch einige überflüssige. Durch die Minimap, welche in Abbildung 4-40 noch einmal vergrößert dargestellt ist, kann man sich zu besseren Orientierung ein Bild der Umgebung machen. Hierfür zeigt ein Pfeil in die Richtung, in die man gerade schaut.

Abbildung 4-40: Minimap im Detail

4.2 Teammitglied „Andreas Zammernig“

4.2.1 Aufgabenstellung

Pflichtenheft

- Erstellen des Pflichtenheftes gemeinsam mit der Projektpartnerin

Einüben in die Entwicklungsumgebung

- Einlesen in Unity3D, sowie Kennenlernen der Integrationsmöglichkeit der Kinect.
- Abschätzung und Besprechung des Spielkonzeptes unter Berücksichtigung der Recherchen zu sinnvollen und effizienten Lernmethoden zusammen mit der Projektpartnerin.

Einbindung der Kinect

- Integration der Kinect in die Entwicklungsumgebung von Unity3D mit der Microsoft-SDK.
- Sicherstellung der Kommunikation zwischen Kinect und Anwendung.
- Erstellung diverser Testlevel zur Verifizierung.
- Entwicklung zusätzlicher Aufgaben, welche nur auf Kinect basieren und dem Nutzer eine natürliche und interaktive Steuerung in der Spielumgebung ermöglichen.

Gemeinsame Levelgestaltung

- Absprache des generellen Levelaufbaus gemeinsam mit der Projektpartnerin, um eine individuelle Einzelarbeit mit anschließender Kombination zu erreichen.

Recherchen zu Legasthenie

Durchführung diverser Recherchen zu dem Thema Legasthenie. Die Nachforschung sollte beinhalten:

- Behandlungsmöglichkeiten
- Häufig anfallende Rechtschreib- und Leseschwächen
- Beliebteste Lernspiele mit größtem Lernerfolg

Dokumentation

- Dokumentation lt. Anforderung der HTL Mössingerstraße

4.2.2 Grundlagen und Methoden

4.2.2.1 Legasthenie

4.2.2.1.1 Definition

„Ein legasthener Mensch, bei guter oder durchschnittlicher Intelligenz, nimmt seine Umwelt differenziert anders wahr, seine Aufmerksamkeit lässt, wenn er auf Symbole, wie Buchstaben oder Zahlen trifft, nach, da er sie durch seine Teilleistungen anders empfindet als nicht legasthene Menschen. Dadurch ergeben sich Schwierigkeiten beim Erlernen des Lesens, Schreibens oder Rechnens.“ [EOEa 95]

„Die Legasthenie ist eine umschriebene und schwerwiegende Beeinträchtigung des Erlernens von Lesen und Rechtschreibung, die in Besonderheiten von Herzfunktionen begründet ist. Diese in allen Schriftsprachen vorkommende Teilleistungsstörung ist veranlagt und nicht die Folge von unzureichender Intelligenzminderung oder anderen körperlichen, neurologischen oder psychischen Erkrankungen.“ [WARN 02]

Es gibt viele Begriffe, welche als Synonym für Legasthenie verwendet werden. Folgende Liste zeigt eine Vielzahl von solchen Begriffen:

- Leseschwäche
- Rechtschreibschwäche, Rechtschreibstörung
- Dyslexia
- Umschriebene Entwicklungsstörungen
- Teilleistungsschwäche
- Phonologisch-phonematische Schwäche
- Lernschwäche
- Lese-Rechtschreibschwäche
- Lese-Rechteschreibstörung [KIP 13]

4.2.2.1.2 Ursache

LRS (Lese- und Schreibschwäche) ist eine im Laufe der Zeit erworbene Schwäche, welche eine sehr ähnliche Problematik wie Legasthenie hat. Dadurch entsteht oft der Irrtum, dass auch Legasthenie durch bestimmte Lebensereignisse hervorgerufen wird.

Tatsächlich ist aber, die Ursache für Legasthenie bei der Genetik der Kinder zu suchen. Aufgrund ihrer Genetik kann eine Person differente Sinneswahrnehmung hervorrufen.

Es gibt dafür aber auch keinerlei äußere Kennzeichen, wie zum Beispiel Hör- oder Sehprobleme. [WARN 02]

4.2.2.1.3 Beispiel

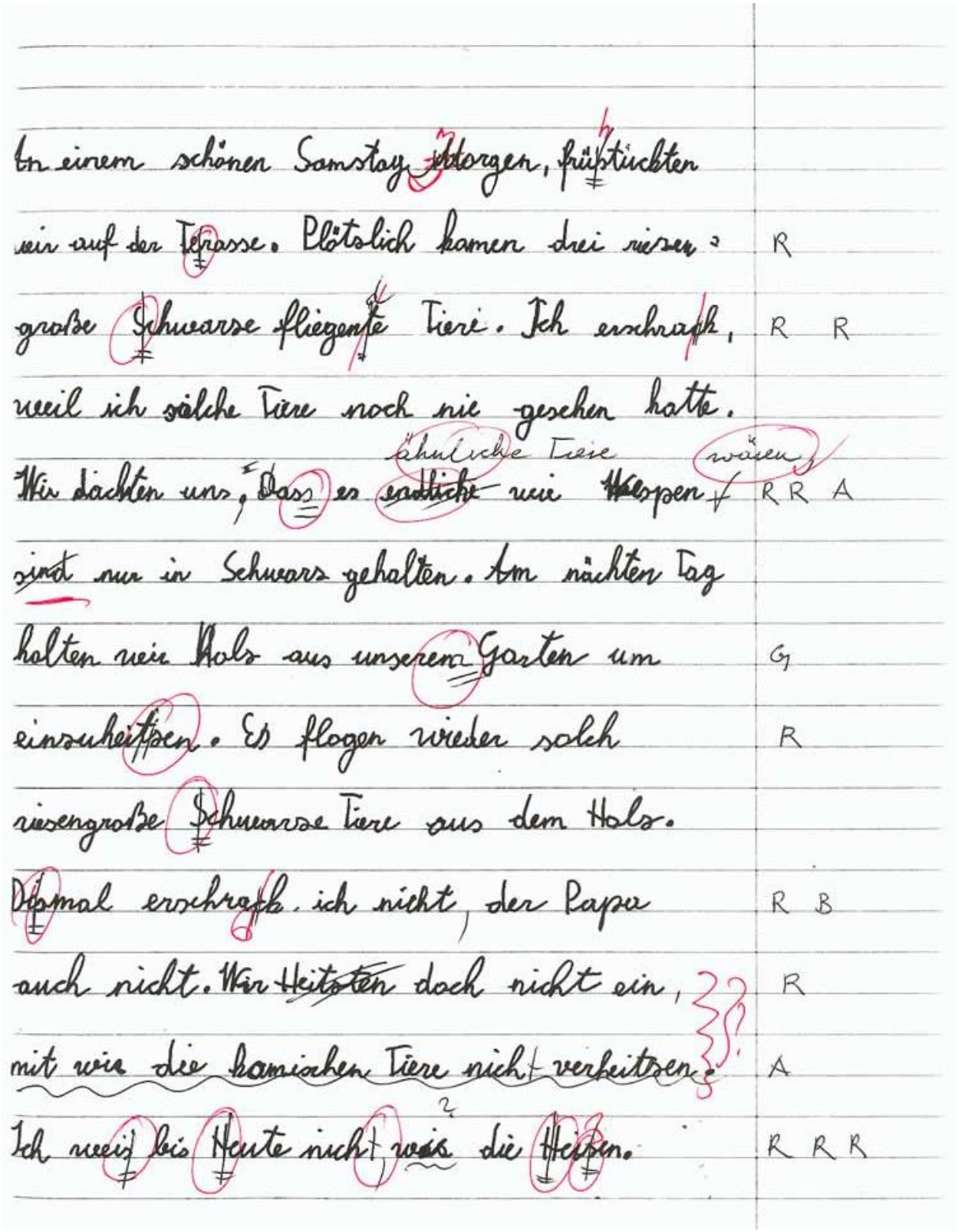

In einem schönen Samstag Morgen, frühstückten
wir auf der Terasse. Plötzlich kamen drei riesen R
große Schwarze fliegende Tiere. Ich erschrafk, R R
weil ich solche Tiere noch nie gesehen hatte.
Wir dachten uns, dass es ähnliche Tiere wären R R A
sind nur in Schwarz gehalten. Am nächten Tag
holten wir Holz aus unserem Garten um G
einzuheitzen. Es flogen wieder solch R
riesengroße Schwarze Tiere aus dem Holz.
Diesmal erschrafk ich nicht, der Papa R B
auch nicht. Wir Heitzten doch nicht ein, R
mit wie die komischen Tiere nicht verheitzen? A
Ich weiß bis Heute nicht, was die Heißen. R R R

Abbildung 4-41: Beispiel für Legasthenie [BSP 15]

Als Beispiel für einen Text des von einem unter Legasthenie leidenden Kind ist in Abbildung 4-41 zu sehen. Hier sind besonders einige der typischen Fehler die bei legasthenen Kinder auftreten können.

4.2.2.1.4 Erscheinungsbild

Legasthenie macht sich oft schon zu Beginn des Schriftspracherwerbs erkenntlich und deshalb sollte auch in der frühen Entwicklung auf Anzeichen geachtet werden. Legasthenie liegt vor, wenn es Probleme beim Umgang mit Symbolen gibt. Insbesondere, wenn es sich um Buchstaben handelt. Durch die Lernschwäche kann es sein, dass sich Aufmerksamkeitsprobleme bei dem Kind äußern, da es sich nicht beachtet und mit großen Problemen konfrontiert sieht. Trotz der auftretenden Schwierigkeiten beim Umgang mit Symbolen verfügt ein Kind, das unter Legasthenie leidet, über normale bis überdurchschnittliche Intelligenz. [WARN 02]

Anzeichen von Legasthenie

Beim Wahrnehmen von Symbolen zum Beispiel beim Sehen und Hören, hat ein Kind mit Legasthenie Schwierigkeiten. Wenn es dann darum geht, das gehörte oder gesehene Symbol sich zu merken und später wiederzugeben, hat eine von Legasthenie betroffene Person Probleme.

Auch bei der grafischen Darstellung von Buchstaben beziehungsweise Symbolen im Allgemeinen gibt es Probleme. So ist das Kind leicht ablenkbar von der eigentlichen Aufgabe.

Da sich das Kind beim Arbeiten mit Buchstaben und Zahlen schwer tut, dauern Arbeiten wie Schreiben, Lesen und Rechnen durchschnittlich länger als andere Kinder aus der Klasse.

Eine Person, die unter Legasthenie leidet, führt oft unkoordinierte Bewegungen aus. Diese betreffen nicht nur die Grobmotorik sondern auch die Feinmotorik.

Das Schriftbild eines legasthenen Kindes ist meist schwer leserlich. Es hat Probleme beim Einhalten von Rändern und Linien, da Schwierigkeiten mit der Raumorientierung beim Schreiben und Lesen auftreten.

Wegen der Orientierungsschwierigkeiten und da sich Kinder mit Legasthenie beim Umgang mit Symbolen schwer tun, kommt es beim Lesen oft zum Stocken. Dabei werden Wörter oft ausgelassen oder oftmals auch Wörter vorgelesen, welche im zu lesenden Text gar nicht vorkommen.

Legasthenie macht sich vor allem im Deutschunterricht bemerkbar. Beim Schreiben und Lesen gibt es häufig Fehler und Probleme mit der Sprache. Folgende Fehler sind dabei häufig zu finden:

Reihenfolgefehler - Die Reihenfolge der Buchstabe innerhalb eines Wortes wird umgestellt. So wird zum Beispiel der zweite und dritte Buchstabe eines Wortes miteinander vertauscht

Reversionen – Reversionen sind Verdrehungen von Buchstaben. Das heißt, es wird anstelle eines „b“ ein „d“ oder anstatt „p“ ein „q“ geschrieben.

Auslassungen – Buchstaben innerhalb eines Wortes werden einfach weggelassen.

Verwechslungen – Verwechslung von ähnlich klingenden Buchstaben wie „d“ und „t“ oder „g“ und „k“.

Dehnungsfehler – Die Dehnung von Wörtern wird außer Acht gelassen. Das heißt, es wird zum Beispiel ein „stummes h“ weggelassen.

Regelfehler - Die Groß- und Kleinschreibung der Wörter ist fehlerhaft.

Einfügungen – Bei Wörtern werden falsche Buchstaben eingefügt, welche nicht Teil des eigentlichen Wortes sind.

Bei nicht von Legasthenie betroffenen Schülern können viele dieser Anzeichen beziehungsweise Fehler ebenfalls auftreten. Der Unterschied ist aber, dass bei Personen mit Legasthenie diese Fehler immer wieder auftreten aber zwischenzeitlich auch wieder richtig geschrieben und es wird dasselbe Wort immer unterschiedlich falsch geschrieben. [WARN 02]

4.2.2.1.5 Behandlung

Wenn bei einem Kind Legasthenie festgestellt wird, müssen gegen die Legasthenie entsprechende Maßnahmen gesetzt und eine Behandlung angesetzt werden. Eine solche Behandlung kann sich einerseits unterteilen in ein Training und eine Förderung gegen die Schwächen von unter Legasthenie leidenden Kindern als auch eine psychologische Behandlung.

Klassische Übungen und die Bekämpfung der häufigsten Fehler stellen nur einen geringen Bereich des Trainings dar. Es muss auch noch zusätzlich die Aufmerksamkeit und Sinneswahrnehmung des Kindes gefördert werden. Es ist wichtig, dass nicht allgemeine Lese- und Schreibübungen vom Kind vollzogen werden, sondern stattdessen jene, die auf von Legasthenie betroffene Personen abgestimmt sind.

Auch muss es bezüglich der Schule Förderungen geben, wie zum Beispiel Deutschnachhilfe und Förderkurse in all jenen Bereichen, wo das Kind aufgrund von Legasthenie Defizite hat. Es kann zusätzlich die Möglichkeit in Betracht gezogen werden, dass aufgrund der Legasthenie das Kind in gewissen Fächern eine andere Benotung erhält.

Für eine möglichst erfolgreiche und zielführende Therapie muss diese gut und über einen größeren Zeitraum geplant werden.

Zu Beginn steht eine Diagnose der Schwachstellen und Stärken des Kindes und ob beziehungsweise auf welche Weise Legasthenie das Kind zusätzlich belastet hat.

Die Ergebnisse der Diagnose müssen anschließend mit dem Kind selbst, den Erziehungsberechtigten und den Lehrpersonen erklärt und besprochen werden. Des Weiteren wird darüber beraten, wie die folgende Therapie organisiert und durchgeführt wird. Hierzu zählen die genauen Therapiemethoden, -übungen und die Hilfsmaßnahmen.

Wenn die Probleme des Kindes zu gravierend werden und ein schulisches Versagen droht, sollte eine ebenfalls noch eine psychologische Behandlung erfolgen.

Generell sollte auf die von Legasthenie betroffene Person besonders rücksichtsvoll eingegangen werden. So sollte man sich über die Stärken und Schwächen von Legasthenieerkrankten bewusst sein und versuchen, dass kein Frust oder Resignation aufkommt. Rückhalt, Geduld und Lob werden besonders vom Kind gebraucht. [KIJ 14] [EOEb 95]

4.2.2.2 Natural User Interfaces

4.2.2.2.1 Definition

Es gibt drei Generation von Benutzerschnittstelle, welche ihre Anwendung finden: Command Line Interfaces (CLI), Graphical User Interfacs (GUI) und Natural User Interfaces (NUI).

Als *Natural User Interface* bezeichnet man Interfaces, welche über natürliche und intuitive Interaktion gesteuert werden. Dazu können zum Beispiel die Sprache oder Gesten verwendet werden. Das bedeutet es wird auf intuitive Fähigkeiten zurückgegriffen, welcher der Benutzer bereits in seinem Leben erlernt und eingesetzt hat. Das Ziel eines NUI ist es auch immer die Komplexität möglichst gering zu halten. Es sollte leicht bedienbar sein, auch ohne die Bedienungsanleitung vorher gelesen zu haben.

Ein Beispiel für ein solches Interagieren kann der Touchscreen oder die Sprachsteuerung eines Smartphones. Auch die KINECT ist ein Vertreter der intuitiven und natürlichen Kommunikation mit einem Gerät. [VGN 10]

4.2.2.2.2 Technologie hinter Natural User Interfaces

Die Technologie von NUIs basiert auf folgenden drei Komponenten: Sensoren, Komparatoren und Aktoren.

Die Sensoren dienen zum Erfassen der Interaktionen des Nutzers. Dabei ist zu beachten, dass nicht jede Bewegung einer Person gewollt ist, sondern es auch Bewegungen gibt, welche man nicht erfassen sollte. Aus diesem Grund ist zu beachten, wie groß die Empfindlichkeit und die Reichweite der Sensorik sind.

Die Komparatoren vergleichen den Zustand vor und nach der Interaktion des Users. Aufgrund des Vergleiches entscheidet der Komparator, welche Aktion durch die Interaktion ausgelöst werden sollte.

Die Aktoren führen dann die gewünschte Aktion, welche von den Komparatoren ausgewählt wurde, aus.

Komparatoren und Aktoren sind in Form von Software realisiert und die Sensoren sind in Hardware ausgeführt. [WIT 14] [KFW 14]

4.2.2.2.3 OSIT Model

Das OSIT Modell gliedert sich in vier Schritte. Beim Design von NUI kann dieses Modell herangezogen werden, wenn der Nutzer mit dem Programm interagieren soll.

orientieren → selektieren → informieren → transagieren

Orientieren

Bei diesem Schritt verschafft sich der/die NutzerIn einen Überblick über das gesamte Angebot. Es geht hierbei vor allem darum, ihm eine möglichst gute Übersicht zu gewährleisten. Details sind dabei nicht nötig, sondern es sollte nur die Grundidee an den Nutzer gelangen. Aussagekräftige Symbole und Shortcuts sind gefragt. Auch eine Wahl der Ansicht

beziehungsweise Darstellung der Gesamtübersicht hilft dem Nutzer, die für ihn geeignete Perspektive und Visualisierung zu wählen und er kann sich somit leichter orientieren.

Selektieren

Die Auswahl eines Objektes sollte für den Nutzer möglichst natürlich und direkt sein, denn dies sind ja die Merkmale eines NUI. Hier kann die KINECT eingesetzt werden um per Gesten ein Element auszuwählen. Das ausgewählte Element sollte mittig am Bildschirm platziert und hervorgehoben werden und es sollte klar ersichtlich sein, was ausgewählt wurde. Je natürlicher der Vorgang wirkt, desto angenehmer ist es für den Nutzer.

Informieren

Beim Informieren sollte das Objekt, über welcher man sich vertraut machen will, im Fokus stehen. Es kann ruhig den ganzen Bildschirm ausfüllen. Der Rest sollte dabei verblassen und in den Hintergrund treten. Detailinformationen und eine detailreichere Ansicht sind hier erwünscht.

Transagieren

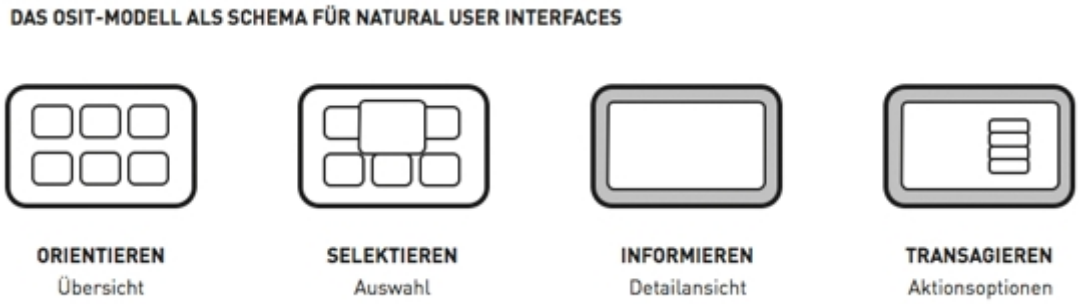


Abbildung 4-42: Blockschaltbild OSIT-Modell [OSI 15]

Beim Transagieren werden die Aktionsoptionen für das zuvor gewählte Element gelistet. Das bedeutet es wird dargestellt, was man mit dem Objekt nun tun kann. Wie bei den anderen Schritten sollte auch hier auf eine natürliche und intuitive Weise, die Optionen ausgewählt werden können. [NUI 13]

In Abbildung 4-42 sieht man die schematische Darstellung, die einzelnen Schritte der Vorgangsweise nach dem OSTI-Modell. Zuerst erfolgt das Orientieren anhand einer Übersicht. Als Nächstes erfolgt das Selektieren eines Elementes aus einer Auswahl. Nach dem Schritt des Auswählens kann sich die Person über das Element per Detailansicht informieren. Als Letztes muss die Person noch entscheiden, was mit dem bereits ausgewählten Element getan werden sollte.

4.2.2.3 Kinect

4.2.2.3.1 Beschreibung

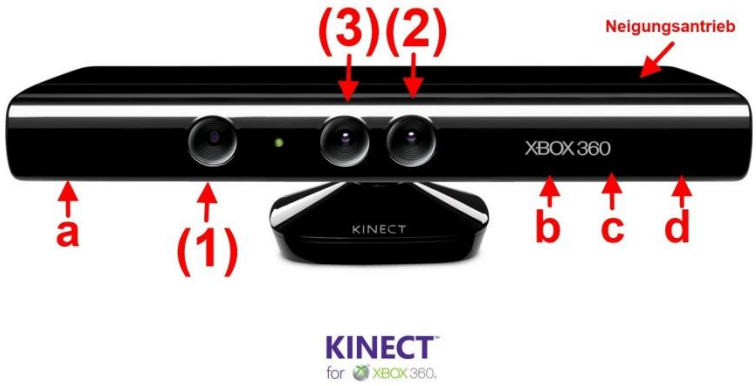


Abbildung 4-43 Kinect [KIN 15]

Kinect ist eine Hardware, die ursprünglich für die Spielekonsole XBOX unter dem Projektname Natal entwickelt wurde. Mit ihr ist es möglich mit der Spielekonsole zu interagieren. Sie basiert auf folgenden Komponenten PrimeSense-Tiefensensor, 3D-Mikrofon, Farbkamera und zugehöriger Software. Weil mit ihr die Sprach oder Gesten zur Steuerung verwendet werden, ist sie optimal für ein Natural User Interface. Die Steuerung per Sprache ist aber noch nicht sehr ausgereift und wird nur zusätzlich zur Steuerung durch Bewegung eingesetzt. Die Kinect wird besonders von Microsoft unterstützt und genutzt. Deshalb ist es am geeignetsten sie auf einem Windows-Betriebssystem zu installieren und zu verwenden, da hierbei die Verwendung einfach ist und zusätzliche Funktionen verwendet werden können.

4.2.2.3.2 Komponenten

Die Kinect basiert auf folgenden fünf Komponenten:

- Multi-Array Mikrofon (a,b,c,d)
- Infrarot-Projektor (1)
- RGB-Kamera (3)
- Infrarot-Tiefensensoren (2)
- Motor zum Neigen der KINECT (Neigungsantrieb)

Multi-Array Mikrofon

Es befinden sich vier einzelne Mikrofone verteilt im oberen Teil der Kinect. Jedes dieser Mikrofone hat einen 16 Bit Mikrofonkanal mit einer Abtastfrequenz von 16 kHz als Ausgang. Die Audio wird von einem Mikrokontroller verarbeitet.

Dadurch ist eine Spracherkennung durch den Benutzer möglich.

Infrarot-Projektor

Der Infrarot-Projektor sendet Infrarotstrahlen, welche dann vom Nutzer und der restlichen Umgebung vor der Kinect reflektiert werden. Infrarotlicht wird mit einer Wellenlänge von 780nm ausgestrahlt. Dies geschieht in Form von vielen kleinen Lichtpunkten, welche gitterförmig angeordnet sind. Diese werden nach der Reflektion vom Tiefensensor empfangen. Da das Empfangen des Infrarots auch ohne externe Lichtquelle möglich ist, funktioniert das Erkennen des Nutzers und seiner Gesten auch bei schlechten Lichtverhältnissen.

RGB-Kamera

Die RGB-Kamera liefert ein Bild mit einer 8Bit VGA-Auflösung von 640*480 Pixel und ist ein CMOS Sensor. Das Sichtfeld der Kamera beträgt horizontal 57° und vertikal 43°, welches bei dem Tiefensensor die selben Maße hat.

Infrarot-Tiefensensor

Der Infrarot-Tiefensensor besteht aus einem monochromen CMOS Sensor, welcher das reflektierte Infrarotlicht empfängt. Der Sensor liefert einen Videostream in VGA-Auflösung, diese ist 640*480 Pixel und hat eine Tiefe von 11 Bit, welcher 2048 Tiefenstufen ermöglicht. Es kann auch der Videostream direkt übertragen werden, das heißt bevor er zum Beispiel zu einer Tiefenkarte konvertiert wurde. In der Tiefenkarte erscheinen nahe Flächen in weiß, welche je nach Distanz in einer anderen Farbe dargestellt werden. Die Farbe für die am weitesten entfernten Flächen ist blau.

Neigungsmotor

Um die Sensor der Kinect neigen zu können, besitzt diese einen Motor, welcher zwischen dem Sockel und dem oberen länglichen Teil der Kinect befindet. Dieser kann jeweils 27° nach oben und unten geneigt werden. So können die Sensoren der Kinect, je nach Platzierung der Kinect, immer auf den Nutzer ausgerichtet werden.

4.2.2.3.3 Interne Verschaltung

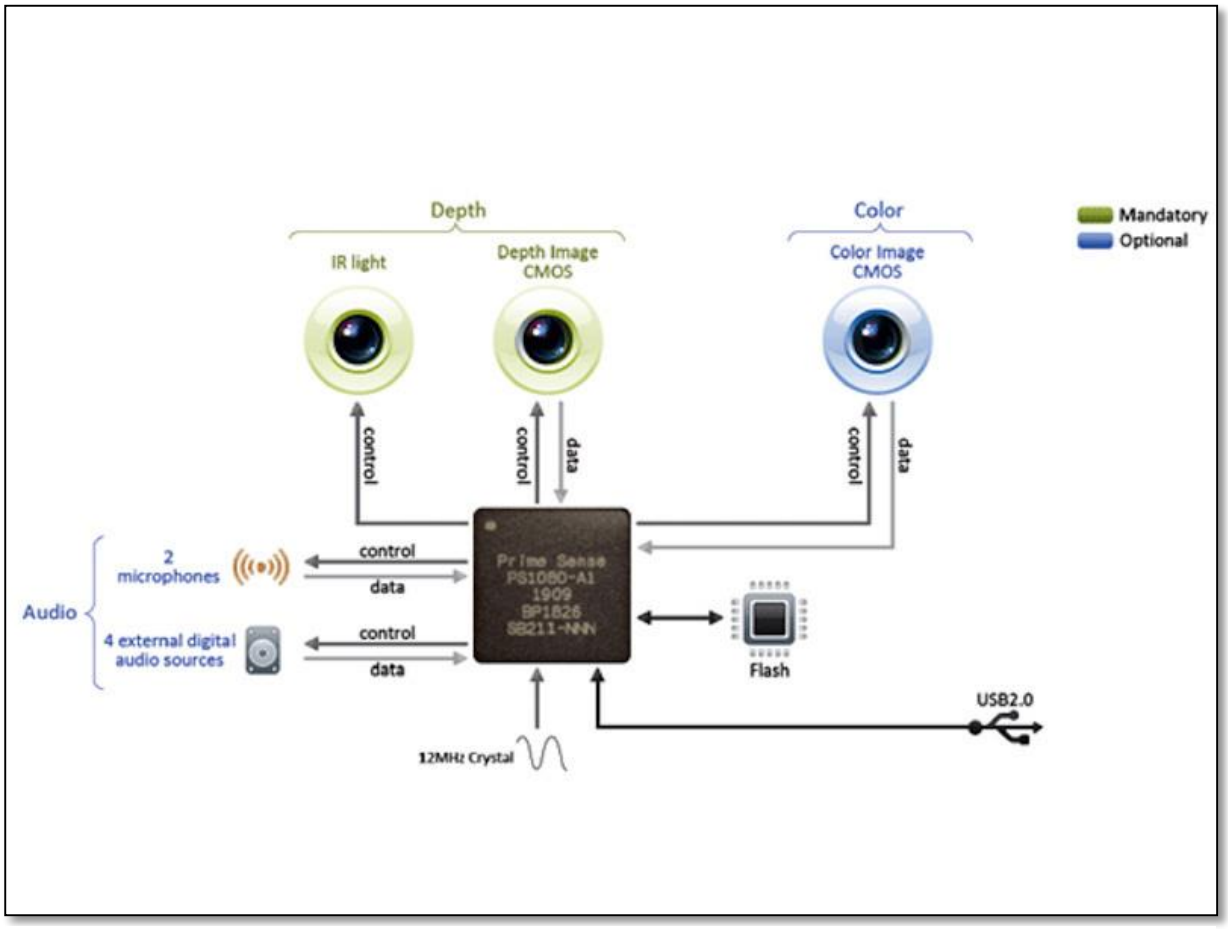


Abbildung 4-44: Interne Verschaltung Kinect [BLS 15]

In der Abbildung 4-44 ist die interne Verschaltung der Kinect zu sehen:

Das Verbindungsstück zwischen Kinect und dem PC bildet die USB- Schnittstelle. Per Kabel wird die Kinect an einem freien USB-Slot am PC angesteckt. Über das Kabel werden alle benötigten Daten übertragen.

Für die Erfassung der Audiosignale sind die Mikrofone zur Linken der Grafik. Zu ihnen führen sowohl Daten- als auch Steuerleitungen. Es werden die einzelnen vier Signale zu einem gesamten Audiosignal zusammengeführt. Diese Aufgabe übernimmt der Chip.

Für das Tiefenbild sind der Infrarot-Projektor und der Infrarot-Sensor zuständig. Der Projektor projiziert nur die Infrarotstrahlen, deshalb führt nur einen Kontrollleitung zu ihm. Der Sensor hingegen erfasst die reflektierten Infrarotstrahlen und ermittelt damit Bilddaten. Er verfügt über Daten- und Kontrollleitung.

Um normale Bilddaten zu empfangen, besitzt die Kinect den CMOS-Sensor, welcher ebenfalls wie der Infrarot-Sensor mit dem Chip verbunden ist. Er liefert alle nötigen Bilddaten. Für die Gestenerkennung allein sind die Bilddaten aber nicht notwendig, sondern nur das Tiefenbild, welches mit dem Infrarot-Projektor und -Sensor ermittelt wird.

Das Herzstück bildet der Chip, welcher von der Firma PrimeSense angefertigt wurde. Dieser befindet sich in der Mitte der Grafik. Er verarbeitet die erfassten Daten und leitet diese dann weiter an den PC über das USB-Kabel.

Damit der Chip Daten zwischenspeichern kann, ist er mit einem Flash-Speicher verbunden. Bei der Errechnung der Bild- und Audiodaten wird dies benötigt.

Für die interne Synchronisation des PrimeSense-Chips braucht dieser eine Taktvorgabe. Durch den an ihn angeschlossenen Quarzkristall, der eine Frequenz von 12MHz liefert, wird dies sichergestellt.

4.2.2.4 Microsoft Kinect - Microsoft SDK

Um eine Kommunikation zwischen Kinect und Unity3D oder Windows herzustellen, wird das Microsoft SDK verwendet. Das SDK wurde von Microsoft entwickelt und kann auch auf der offiziellen Seite von Microsoft heruntergeladen werden.

Um eine Kommunikation zwischen Kinect und Windows herzustellen, wird das Microsoft SDK verwendet. Das SDK wird von Microsoft für alle Nutzer und Entwickler kostenlos zur Verfügung gestellt und kann von der offiziellen Seite von Microsoft heruntergeladen werden. Es gibt mehrere Versionen des SDKs, wobei die allerneueste, welche die Version 2.0 ist, erst als Beta-Version gibt. Die Version 1.8 ist die aktuellste Version, welche als komplettes Release von der Seite heruntergeladen werden kann. Es gibt zwar bereits die Version 2.0, diese befindet sich aber zum derzeitigen Zeitpunkt noch in der Beta-Phase Die Wahl der richtigen Version des SDKs hängt von der vorhandenen Hardware und Software ab. Je neuer ein SDK ist, desto mehr Funktionen können genutzt werden, auch Verbesserungen gegenüber älteren Versionen sind mitinbegriffen.

Parallel zum SDK gibt es noch ein Developer Toolkit, welches für Entwickler angedacht ist. Dieses ermöglicht es, dem Nutzer selbst Programme und Projekte zu entwickeln, welche die Kinect einbinden. [SDK 14]

4.2.2.4.1 Inhalt

Im Downloadumfang des SDK sind mehrere Dateien enthalten. Eine Dokumentation über das SDK ist mit dabei. Mehrere Beispielprogramme sind auch inkludiert. Diese sind in folgenden Programmiersprachen ausgeführt: C#, C++, und Visual Basic. Sie können gleich direkt an der Kinect getestet werden und es kann in die jeweilige Dokumentation des Programmes Einsicht genommen werden. Dies eignet sich als Funktionstest der Kinect. Header Files für eigene Programme und Libraries, welche ebenfalls ins eigene Programm eingebunden werden können stehen auch zur Verfügung mit der SDK. Es können noch viele weitere Komponente und Tools über die SDK heruntergeladen werden. [SDK 14]

4.2.2.5 Game-Design

4.2.2.5.1 Einführung

Um ein Spiel zu entwickeln braucht man einerseits eine gewisse Technologie, damit es programmiert werden kann, es muss aber ebenfalls eine Spielidee vorhanden sein und diese sollte gut umgesetzt werden.

Beim Game-Design geht es um alle Aspekte des Spieles. Es betrifft das Design des einzelnen Levels, aber auch das Spielkonzept im Allgemeinen. Es darf der spielenden Person nicht langweilig werden und der/die SpielerIn sollte gefordert werden. Besonders bei Lernspielen ist es wichtig, sich nicht nur auf den Lerneffekt zu konzentrieren, sondern auch das Kind beim Spielen zu unterhalten, sodass es das Spiel überhaupt erst spielen will. Es ist die Aufgabe eines Game Designer aus dem Lernprogramm ein Lernspiel zu machen, welches auch spielerische und spannende Aspekte beinhält. Um folgendes zu erreichen, muss auf die Zielgruppe eingegangen werden.

Es ist seit einigen Jahren möglich Game-Design als Lehrberuf oder Studium zu erlernen. [GDTP 04]

4.2.2.5.2 Spielmechanik

Im Zuge der Spielmechanik werden sogenannte Spielregeln festgelegt, die im Spiel vorherrschen. Die Spielmechanik beschreibt im Allgemeinen den Ablauf des Spieles. Es sollte aus einer gewissen Ausgangssituation eine oder mehrere Aufgaben erfüllt werden. Bei diesem Prozess sollte ein Spielerlebnis entstehen. So gut wie alle Aspekte des Spiels werden hierbei berücksichtigt. Die Interaktion mit der spielenden Person, interne Abläufe und die Fortentwicklung der Spielwelt müssen behandelt werden.

Spieler wollen gefordert werden, möchten aber nicht mit einer ausweglosen Situation konfrontiert werden. Die Aufgaben in einem Spiel sollten klar verständlich und logisch nachvollziehbar sein. Je nach Zielgruppe muss die Schwierigkeit und Komplexität der Aufgaben variiert werden. Bei zu erfüllenden Aufgabenstellungen sollte Freiraum zur Lösung der Problemstellung geschaffen und nicht nach einem strikten Muster vorgegangen werden. Die Person wird somit gefordert, kann eine eigene Lösung entwickeln und muss nicht zwingend den Vorgaben folgen, sondern kann zwischendurch die Spielumgebung erkunden oder vorherige Level erneut spielen. [GDTP 04]

4.2.2.5.3 Spielwelt

Die Spielwelt sollte ebenfalls, wie die Spielmechanik, auf die entsprechende Zielgruppe abgeglichen werden. Besonders bei Kinder sollte die Welt freundlich und einladend wirken. Als Spielwelt wird all jenes bezeichnet, was den Spieler umgibt. Dies umfasst das Design der interaktiven Spielobjekte bis hin zum Design der passiven Spielumgebung. Man sollte sich in der Spielwelt frei bewegen können und man darf sich nicht zwischen Spielobjekten verfangen oder die Spielwelt ungewollt verlassen. Es sollte der Spielwelt eine Grenze gesetzt werden, welche aber nicht allzu offensichtlich sein und natürlich erscheinen sollte. So kann die spielende Person die Spielwelt erkunden und es wird das Gefühl vermittelt, in einer eigenen Welt zu sein

Das Design der gesamten Spielwelt sollte einheitlich sein und ineinander übergehen. Wenn eine Geschichte im Spiel behandelt wird, sollte natürlich die Spielwelt passend zu dieser gewählt werden. Passive Elemente, wie sich bewegende Tiere oder Wolken, lassen die Spielwelt natürlicher und spannender erscheinen, es sollte aber mit solchen Elementen nicht übertrieben werden. [GDTP 04]

4.2.3 Realisierung

4.2.3.1 Buchstaben sammeln

Beim Level geht es darum die Buchstaben eines Wortes in richtiger Reihenfolge einzusammeln. Am Anfang wird dem Kind das Wort gezeigt und sobald es die Eingabetaste drückt, sieht es nur noch die Buchstaben, die es einsammeln muss.

Die spielende Person kann einen Avatar steuern um die Buchstaben einzusammeln. Als Grundlage dient der First Person Controller, welcher als Standard Asset bereits von Unity3D zur Verfügung steht. Durch das Verändern der Kamerapositionierung und durch das Einfügen eines Hasenmodells als Avatar kann der Hase als Avatar gesteuert werden.

Ein Script nimmt zufällig aus einem string-Array ein Wort und zeigt es dem Benutzer, indem er es in einem GUI-Text darstellt. Sobald das Kind die Eingabetaste drückt, wird ein neues Script aktiviert, welches zuvor in der start-Funktion deaktiviert wird. Folgende Code-Zeilen deaktiviert die Kamera, welche auf den GUI-Text gerichtet ist und aktiviert die Kamera, welche den Avatar im Bild hat. Ebenfalls wird das Script, welches für das Einsammeln der Buchstaben verantwortlich ist, aktiviert. Dies geschieht beim Drücken der Eingabetaste.

```
void Update (){
        if (Input.GetKey (KeyCode.Return)) {
            FPSControllerCam.enabled = true;
            mainCamera.enabled = false;
            FPSController.GetComponent<CollectingLetters> ().enabled = true;
        }
    }
```

Das Script, welches aktiviert wird, verteilt zufällig die Buchstaben des zuvor ausgewählten Wortes auf einem Terrain. Das Terrain ist ein Objekt, welches eine Landschaft darstellt und auch dafür passende Texturen aufweist. Es stellt sozusagen den Boden dar, auf dem sich der Avatar bewegt.

Wenn man Buchstaben von einer anderen Seite betrachtet, vertauscht man diese oft oder erkennt sie nicht. So kann es sein, dass der Buchstabe b als d erkannt wird, wenn er von der falschen Seite betrachtet wird. Um dies zu verhindern und bei der spielenden Person keine Verwirrung verursachen will, besonders wenn diese unter Legasthenie leidet, richtet sich der Buchstabe immer in Richtung des Nutzer aus. Das bedeutet, das Kind sieht den Buchstabe immer von der Vorderseite, da der Buchstabe sich immer nach dem Avatar, also dem Blickwinkel der spielenden Person ausrichtet. Um dies zu gewährleisten, gibt es auf jedem Buchstaben ein Script mit folgender Code-Zeile, welche eben jenes bewirkt:

```
transform.LookAt (GameObject.Find("First Person Controller").transform);
```

Der String “First Person Controller” beschreibt das jeweilige Objekt, auf welches sich der Buchstabe ausrichten sollte.

Um die Buchstaben aufzusammeln muss der Avatar den jeweils nächsten Buchstaben berühren. Das bedeutet, die spielende Person muss mit dem Avatar in den jeweiligen Buchstaben hineinfahren.

4.2.3.2 Schiffeversenken

Dieser Level stellt „Schiffeversenken“ mit Wörtern dar.

Zu Beginn wird ein zweidimensionales string-Array mit den Wörtern befüllt. Es wird ein string-Array verwendet, da die Buchstaben der einzelnen Wörter in das Array gespeichert werden und man die equals-Methode aufrufen kann, welche bei einem Char-Array nicht zur Verfügung stehen würde. Der zuständige Algorithmus schreibt zuerst in das gesamte Array als Platzhalter das Zeichen „.“. Danach beginnt das Einfügen des ersten Wortes. Wörter werden per Zufall vertikal oder horizontal eingefügt. Nach dem Hineinschreiben eines Wortes werden um das Array das Zeichen „x“ als Platzhalter eingefügt, sodass niemals zwei Wörter beieinander ohne Abstand eingefügt werden. Das Einfüllen der Wörter geschieht innerhalb einer while-Schleife. Kann ein Wort nicht eingefüllt werden, da an einer Stelle bereits ein Buchstaben eines anderen Wortes oder der Platzhalter „x“ steht, wird mit der goto-Anweisung und mit continue-Anweisung zum Ende der Schleife, welche dann wieder von Beginn an durchläuft. Parallel zum string-Array existiert ein weiteres Array, welches zur Überprüfung dient.

Danach wird ein Spielfeld erstellt. Dieses weist zwei Spielfelder auf, welche jeweils aus 100 einzelnen Elementen bestehen. Der Inhalt des Spielfeldes wird aus den bereits zuvor erwähnten Array bezogen, in welchem die Wörter gespeichert wurden.

Um leichter ermitteln zu können, ob ein Buchstabe oder ein Platzhalter getroffen wurde und zu welchem Wort der Buchstabe gehört, wenn einer getroffen wurde, werden über dem Spielfeld noch einmal 100 Einheiten erzeugt. Das obere Spielfeld wird mit Hilfe des zweiten Arrays, welches für die Überprüfung da ist, erstellt. Wird auf ein Feld geklickt, dann kann ermittelt werden, ob ein Buchstabe getroffen wurde. Bei einem Treffer kann anschließend ermittelt werden, zu welchem Wort der Buchstaben gehört. Dadurch kann ermittelt werden, ob die spielende Person bereits verloren oder schon gewonnen hat. Zusätzlich kann dadurch festgestellt werden, welche Wörter bereits komplett gefunden wurden.

Die selbst geschrieben Methode createPattern() generiert das Spielfeld. Folgender Ausschnitt zeigt die Erstellung einer Ebene:

```
GameObject spawnedObject = (GameObject)Instantiate (plainObject, new
Vector3 (j, -i, 0), Quaternion.Euler (new Vector3 (0, 90, 0)));
spawnedObject.AddComponent<BoxCollider> ();
spawnedObject.AddComponent<PlainClick> ();
spawnedObject.name = "spawnedObject";
GameObject childObject = GameObject.Find ("/spawnedObject/Cube");
childObject.renderer.material.color = Color.blue;
childObject.AddComponent<GetIndicated> ();
spawnedObject.name = "fertig";
```

Ausgeführt werden die oben erwähnten Code-Zeilen innerhalb von zwei for-Schleifen. Durch die for-Schleifen erhöhen sich die Indexe „j“ und „i“. Dabei werden mehrere Prefabs instanziiert und mit Komponenten, wie dem Script PlainClick und ähnlichem, versehen. Anschließend wird das Child-Objekt des instanziierten Objektes gesucht und die Farbe dessen geändert. Das Child-Objekt bekommt ebenfalls ein Script zugewiesen. Beide Objekte werden auch unbenannt.

Visualisiert wird auch, welche Wörter man bereits aufgedeckt und welche übrig sind. Dies geschieht in Form von vier Symbolen, welche jeweils ein noch nicht gefundenes Wort darstellen. Wird ein Wort gefunden wird das zugehörige Symbol gelöscht.

Als Letztes wurde noch ein Algorithmus geschrieben, welcher einen Gegner verkörpert. Dieser ist in der Lage selbstständig Züge zu vollziehen. Hat der Computergegner einen Buchstaben eines Wortes gefunden, versucht dieser die restlichen Buchstaben des Wortes zu finden. Dadurch wird der Gegner authentischer und es wird für das Kind schwieriger das Spiel zu gewinnen.

4.2.3.3 Levelauswahl

Da das gesamte Projekt ABC-Darian mehrere Levels aufweist, muss es eine Möglichkeit geben, zwischen den einzelnen Level eine Auswahl zu treffen. Um es möglichst ansprechend zu gestalten, wurde bei dem ersten Level besonders auf das Design geachtet.

Abbildung 4-45: Voransicht der Levelauswahl

Für jedes Level, welches im Zuge des Projektes ABC-Darian spielbar ist, stellt einen Punkt entlang eines Weges dar. Die Umgebung wurde mit verschiedensten Objekten versehen, wie Bäume, Steine, Büsche und Pilze, welche den Level authentischer wirken lassen. Um ein Level auszuwählen muss die spielende Person nur den Hasen auf den jeweiligen Wegpunkt setzen. Eine Voransicht ist in Abbildung 4-45.

Die Steuerung des Hasen kann entweder mit den Pfeiltasten „<“ und „>“ oder durch Links- beziehungsweise Rechts-Wischen mit der Kinect ausgeführt werden. Bestätigt wird mit der Eingabetaste beziehungsweise mit der Push-Geste, welche vollführt wird, indem man beide Hände gerade nach vorne hin ausstreckt. Da die Maus bei diesem Level nicht nötig ist und irritieren könnte, wird der Cursor bereits in der start-Funktion durch die nachfolgende Codezeile ausgeblendet.

```
Screen.showCursor = false;
```

Folgende Code-Zeilen beschreiben die Steuerung mit Pfeiltasten und Eingabetaste:

```
if (Input.GetKeyDown (KeyCode.Return)) {
        enter ();
    }
    if (Input.GetKeyDown (KeyCode.LeftArrow)) {
        toPrevious ();
    }
    if (Input.GetKeyDown (KeyCode.RightArrow)) {
        toNext ();
    }
```

Dieselbe Steuerung sieht mit der Kinect folgendermaßen aus:

```
if (slideChangeWithGestures && gestureListener) {
            if (gestureListener.IsSwipeLeft ()) {
                toNext ();
            } else if (gestureListener.IsSwipeRight ()) {
                toPrevious ();
            } else if (gestureListener.IsPushed ()) {
                enter();
            }
        }
```

Wird die enter-Methode aufgerufen, wird der derzeit ausgewählte Level gestartet und der Cursor muss zuvor wieder sichtbar gemacht werden, damit er im nächsten Level zu sehen ist.

Die beiden Methoden toPrevious() und toNext() setzen jeweils den Zielpunkt des Hasen und somit das ausgewählte Level auf einen Wegpunkt nach vorne oder zurück, wenn es möglich ist.

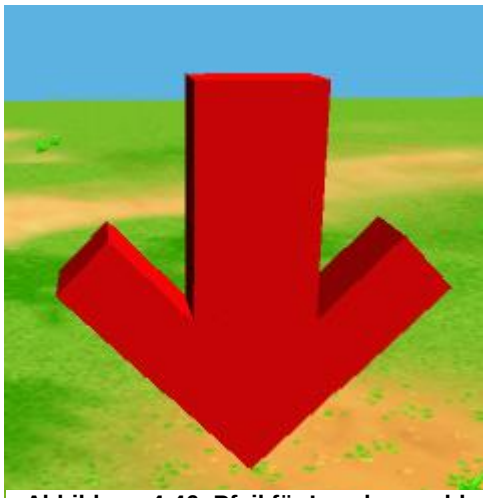


Abbildung 4-46: Pfeil für Levelauswahl

Da der Hase eine gewisse Zeit benötigt um sich springend von Punkt zu Punkt zu bewegen, wird mit einem großen roten Pfeil das aktuelle Level also der aktuelle Punkt auf dem Weg angezeigt. Der Pfeil ist in der Abbildung 4-46 zu sehen.

Um das Springen des Hasen natürlich wirken zu lassen, aber ihn dauernd springen zu lassen, wird unter anderem die lerp-Methode verwendet. Mithilfe dieser Methode bewegt sich der Körper des Hasen von einem Punkt zu einem anderen. Diese Methode, verschachtelt in if-Anweisungen, liefert das gewünschte Ergebnis. Die nachfolgenden Code-Zeilen wurden in die Update-Funktion geschrieben, da diese bei jedem Frame aufgerufen wird und dies für die deltaTime-Funktion nötig ist. Die erste if-Anweisung dient allein dazu erst dann zu hüpfen, wenn der Hase unter einer gewissen Höhe befindet. Dies wird ebenfalls nur benötigt, sodass die Bewegungen natürlicher wirken.

```
if (bunny.transform.position.y > 6) {
            StartCoroutine (moveBunnyTo (wayPoints
[wayPointNumber].transform.position));
        }
        //for jumping
        if (timer > 0 && timer < 1) {
            bunny.transform.position = Vector3.Lerp (bunny.transform.position, bunnyTarget
+ jumpVector, 3 * Time.deltaTime);
        } else if (timer > 1 && timer < 2) {
            bunny.transform.position = Vector3.Lerp (bunny.transform.position, bunnyTarget,
3 * Time.deltaTime);
        } else if (timer > 2) {
            timer = 0;
        }
        timer += Time.deltaTime;

        this.transform.position = new Vector3 (bunny.transform.position.x - 5.5f,
this.transform.position.y, bunny.transform.position.z);
```

Zusätzlich zum Springen gibt es eine Coroutine, welche die horizontale Bewegung des Hasen nachahmt. Es wird daher eine Coroutine eingesetzt, damit diese parallel zum Hauptprogramm ausgeführt wird und somit die beiden Programmteile sich nicht beeinflussen und reibungslos nebeneinander laufen können. Diese Coroutine wird wie andere Methoden außerhalb der update- und start-Funktion geschrieben und sieht wie folgt aus:

```
IEnumerator moveBunnyTo (Vector3 target)
{
        Debug.Log ("" + bunny.transform.position.y);
        bunny.transform.LookAt (new Vector3 (target.x, bunny.transform.position.y, target.z),
Vector3.up);
        bunnyTarget = target;
        Debug.Log ("Bunny moves to " + target);
        yield return new WaitForSeconds (0f);
}
```

4.2.3.4 Buchstaben erraten

Der Level „Buchstaben erraten“ ist dem bekannten Spiel „Hangman“. Es geht darum ein Wort zu erraten anhand des ersten Buchstaben. Anstelle jedes weiteren Buchstaben im Wort steht ein Platzhalter, somit weiß man, aus wie vielen Buchstaben das Wort besteht.

Für jeden Buchstaben, der gewählt wird und nicht im Wort vorhanden ist, wird ein Turm um eine Etage kleiner. Der Turm besteht aus sieben solcher Etagen in verschiedenen Farben und auf der obersten steht ein Hase, welcher hinabfällt, wenn alle Etagen weg sind.

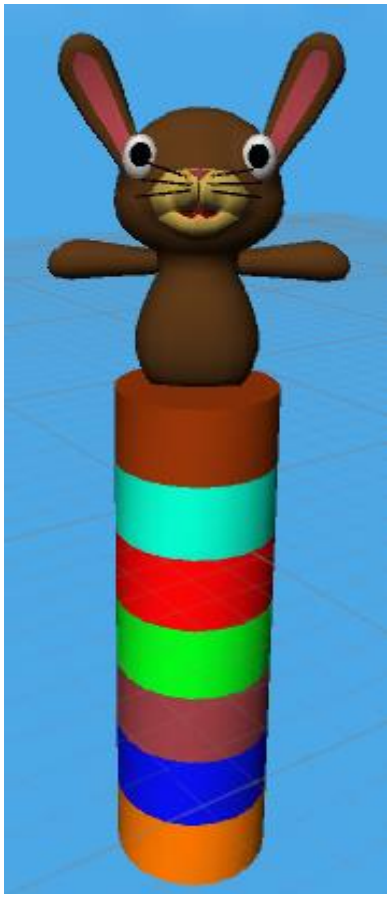


Abbildung 4-47: Modell des Hasen und Turmes

Wenn der Level gestartet wird, erstellt ein Algorithmus den Turm mit Hasen oben drauf. Desweitern wird ein zufälliges Wort aus einer Liste ausgewählt und dargestellt. Die Darstellung erfolgt, wie bereits beschrieben, nur mit dem Anfangsbuchstaben und den Platzhaltern. Um es leichter zu gestalten wird, falls der erste Buchstabe öfter im Wort vorkommt, dieser Buchstabe bereits eingefüllt. In Abbildung 4-47 sieht man das fertige gesamt Modell.

Zu Beginn werden ebenfalls noch alle Buchstaben des Alphabets in Form einer Tastatur erstellt. Man kann auf alle diese Buchstaben klicken. Wird auf einem Buchstaben geklickt, überprüft ein Algorithmus, ob dieser im Wort vorhanden ist. Wenn dieser vorhanden ist, wird er, so oft wie der Buchstabe vorkommt, eingefügt. Ist der Buchstabe nicht vorhanden, wird der Turm um eine Etage verkleinert und das Wort bleibt unverändert. Der geklickt Buchstabe wird danach gelöscht, sodass jeder Buchstabe nur einmal eingefügt werden kann.

Folgende Methode erstellt die virtuelle Tastatur. Mit der instantiate-Methode kann ein Objekt in das Spiel während der Laufzeit eingefügt werden. Die Position und Ausrichtung kann dabei festgelegt werden. Das Objekt kann bereits zuvor mit Scripts, Texturen und anderem versehen werden, welche dann auch im Level erhalten sind. Dafür muss das Objekt nur als Prefab gespeichert werden.

```
private void createKeyboard ()
    {
        int xCoordinate = 10;
        int yCoordinate = -2;
        int zCoordinate = 0;
        for (int i = 0; i < Alphabet.Length; i++) {
            GameObject.Instantiate (Alphabet [i], new Vector3 (xCoordinate, yCoordinate,
zCoordinate), new Quaternion (0, 0, 0, 0));
            GameObject cube = GameObject.CreatePrimitive (PrimitiveType.Cube);
            cube.transform.position = new Vector3 (xCoordinate - 0.5f, yCoordinate + 0.5f,
zCoordinate);
            cube.transform.localScale = new Vector3 (2, 2, 0.5f);
            cube.name = charABC [i].ToString ();
            cube.AddComponent<ClickHelper> ();
            Mesh mesh = cube.GetComponent<MeshFilter> ().mesh;
            mesh.Clear ();

            xCoordinate -= 2;
            if (i == 9) {
                xCoordinate = 9;
                yCoordinate -= 2;
            }
            if (i == 18) {
                xCoordinate = 8;
                yCoordinate -= 2;
            }
        }
    }
```

Abbildung 4-48: Modell der Tastatur

Die Abbildung 4-48 zeigt die erstellte Tastatur. Sie füllt den Großteil des Bildschirmes aus und ist einfach per Mausklick zu bedienen.

4.2.3.5 Wörter abschießen

In diesem Level geht es, darum zu erkennen, ob ein Wort richtig geschrieben ist oder ob ein Wort Rechtschreibfehler enthält. Beim Abschießen eines falsch geschriebenen Wortes erhält man Punkte. Wird ein richtig geschriebenes Wort angeklickt, so verliert man Punkte. Es geht darum möglichst viele Punkte innerhalb einer Minute zu sammeln.

Per Script wird in einem fix vorgegeben Zeitintervall ein Objekt erstellt. Das Objekt selbst besteht aus mehreren Cube Gameobjekten. Ein Algorithmus erstellt diese, indem er ein Wort mit ihnen darstellt. Auf die einzelnen Cubes wird jeweils ein Buchstabe des Wortes gesetzt, so ergeben sich ganze Wörter. Die einzelnen mit Buchstaben versehenen Gameobjekten werden dann zu einem großen gemeinsamen Objekt zusammengefasst.

Das Gesamtobjekt bewegt sich mit einer fix vorgegebenen Geschwindigkeit auf die Kamera zu. Während dieser Bewegung rotiert es zufällig um die eigene Achse. Dadurch zeigen abwechselnd die Seiten der Cubes zum Nutzer. Da die einzelnen Buchstaben auf allen Seiten der Cubes abgebildet sind, kann man das Wort immer erkennen.

Die Methode „InvokeRepeating“ wird von Unity3D selbst zur Verfügung gestellt. Mit ihr kann eine Methode immer wieder für eine gewisse Zeitspanne aufgerufen werden. Das erleichtert das andauernde Aufrufen in der update-Methode. Diese Methode wird einmal in der start-Funktion aufgerufen. Im Verlauf des Spieles sorgt dies dann dafür, dass mit der eigen geschriebenen Methode spawnWords() wird dann immer wieder ein neues Objekt, auf welchem ein Wort abgebildet ist, erstellt.

```
InvokeRepeating ("spawnWord", 2, 2);
```

Um der spielenden Person ein Zeitfenster von exakt einer Minute zu geben, musste eine Art von Timer per Programmcode realisiert werden. Dafür wird in der update-Funktion eine Variable vom Typ Float vermindert. Es wird der Wert jedes Mal der Wert abgezogen dem man aus der Time.deltaTime-Funktion erhält. So wird der Wert immer kleiner, je öfter die update-Funktion aufgerufen wird. Er wird jede Sekunde um eins kleiner. Nach einer Minute erreicht die Variable den Wert null. Dann wird eine Coroutine gestartet, welche die erreichte Punktzahl anzeigt und zur Levelauswahl zurückkehrt.

```
    void Update ()
    {
```

```
        if (time > 0) {
            time -= Time.deltaTime;
            if (time.ToString ().Length == 1) {
                timer.text = " " + time.ToString ("0");
            } else {
                timer.text = time.ToString ("0");
            }
        } else if (time < 0) {
            StartCoroutine (wonGame ());
        }
        scoreTable.text = "" + score;
    }
```

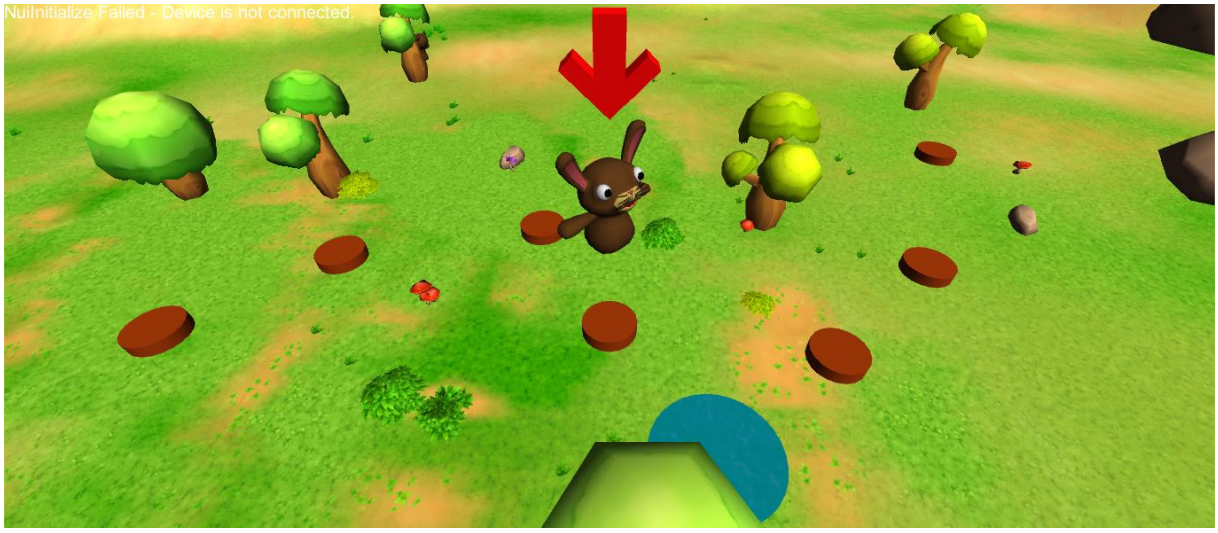


Abbildung 4-49: Spielansicht der Levelauswahl

4.2.4 Ergebnisse

Mit der Levelauswahl ist es möglich zwischen den einzelnen Level jenen Level zu wählen, welchen man gerne spielen würde. Dies wird einerseits bei den Level, welche bei Dyskalkulie helfen, eingesetzt, als auch bei den Level die bei Legasthenie helfen. Der Pfeil zeigt auf den jeweiligen Wegpunkt, welcher den Level repräsentiert und der Hase springt gleich darauf zum selben Punkt und verharrt dort springend. Dies geschieht solange, bis die spielende Person sich für einen anderen Level entscheidet oder den Level bestätigt.

Die Hintergrundobjekte, welche aus Bäumen, Sträuchern, Pilzen, einem See und vielen mehr bestehen, runden das Erscheinungsbild des Levels ab. In Abbildung 4-49 sieht man den Gesamteindruck des Levels.

Das Kind sieht Wörter auf sich zu fliegen. Es hat Zeit sich zu entscheiden, vom Zeitpunkt des Erscheinens des Wortes bis zum Zeitpunkt and dem das Wort wieder verschwindet. Innerhalb dieser Zeitspanne muss die spielende Person sich entscheiden, ob das Wort korrekt ist oder Rechtschreibfehler enthält. Hat das Wort Rechtschreibfehler, so muss es per Mausklick „abgeschossen“ werden. Geschieht dies, erhält das Kind Punkte. Wird ein richtig geschriebenes Wort angeklickt, verliert das Kind Punkte und das Wort verschwindet ebenfalls. Weniger als null Punkte kann aber das Kind nicht erreichen.

Abbildung 4-50: Spielansicht des Shooters

Die blaue Zahl in Abbildung 4-51 auf der rechten oberen Seite des Bildschirmes zeigt die bereits erhaltenen Punkte an. Am rechten oberen Bildschirmrand hingegen sieht man, wie viele Sekunden einem noch bleiben. Wenn diese Zahl den Wert null erreicht hat, endet der Level. In der rechten unteren Ecke des Bildschirmes ist noch ein Button zu sehen, mit welchem der Level vorzeitig beendet werden kann.

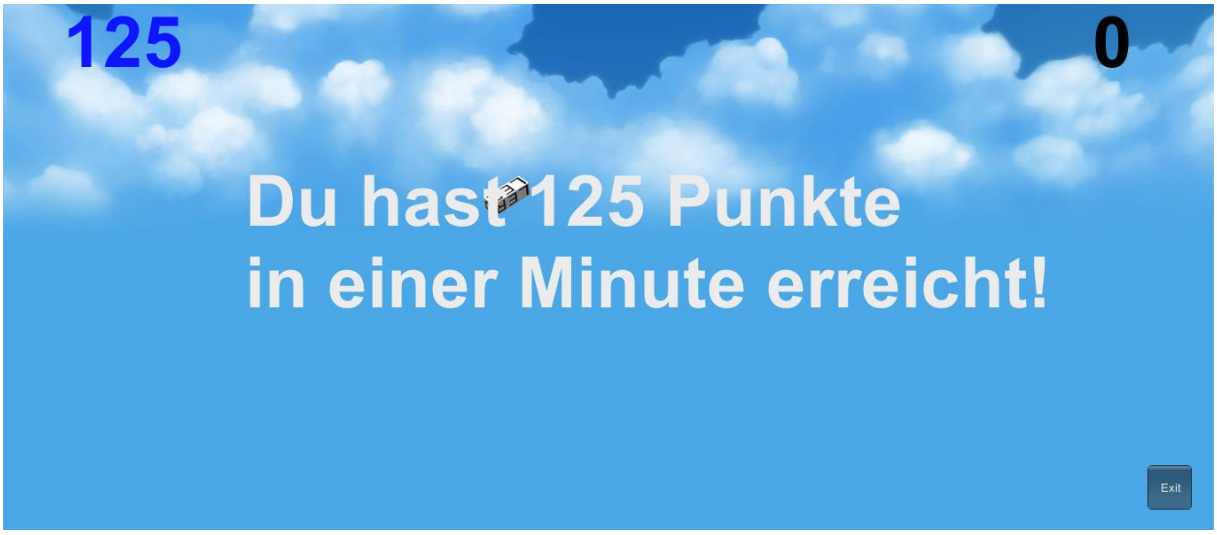


Abbildung 4-51: Ende des Shooter-Levels

Bevor der Level beendet wird, stellt noch ein Textfeld dar, wie viele Punkt die spielende Person in der einen Minute erreicht hat, wie in Abbildung 4-51 zu sehen ist.

In Abbildung 4-52 in schwarzer Farbe dargestellt, sieht man das Wort, welches erraten werden muss. Zur Rechten der Buchstaben sieht man einen Hasen auf einen Turm aus Rollen stehen.

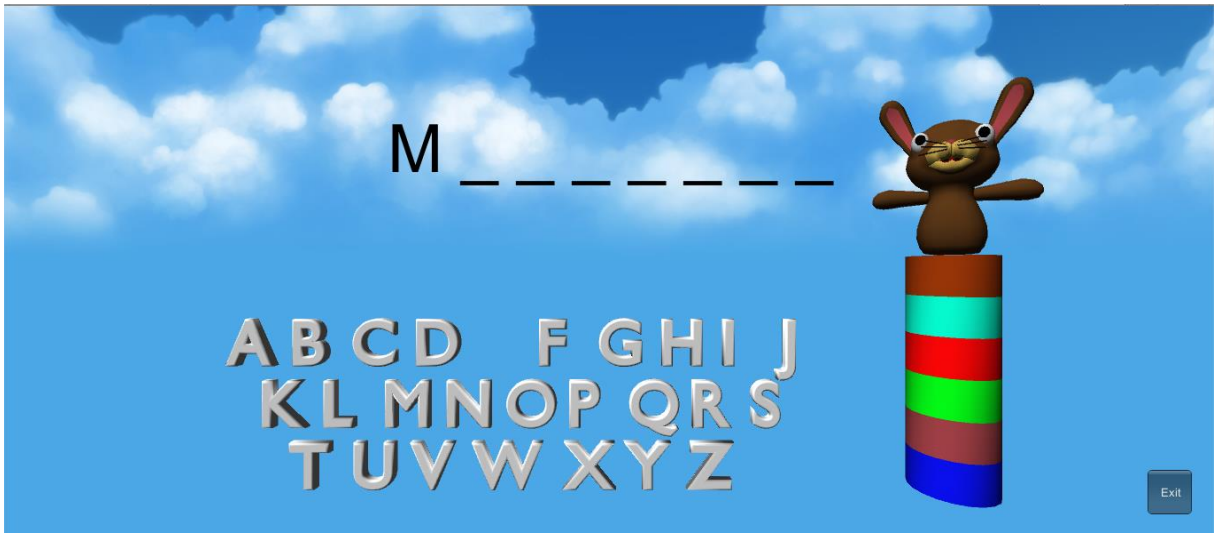


Abbildung 4-52: Spielansicht des Wörter erraten

Zur besseren Darstellung hat jede Rolle eine andere Farbe. Nur die Oberste hat dieselbe Farbe wie der Hase. Wenn diese zerstört wird, dann fällt auch der Hase. Darunter ist das Alphabet in Großbuchstaben dargestellt. Jeder der Buchstaben kann geklickt werden. Wird einer per Mausklickt ausgewählt, wird dieser dann zerstört. Anschließend wird er in das Wort eingetragen, wenn er darin vorkommt. Kommt er nicht im Wort vor, so wird das Turm, auf welchem der Hase steht um eine Etage kleiner. Gewonnen hat die spielende Person, wenn sie das komplette Wort erraten hat. Wird jedoch der Turm zuvor zerstört und der Hase fällt herunter, hat das Kind leider verloren.

Wird das Spiel gewonnen, wird folgender Dialog gezeigt, wie in Abbildung 4-53 zu sehen.

Abbildung 4-53: Wörter erraten - Spiel gewonnen

Wird das Spiel verloren, erscheint der Dialog, welcher in Abbildung 4-54 zu sehen ist.

Abbildung 4-54: Wörter erraten - Spiel verloren

In der Abbildung 4-55 sind die zwei Spielflächen zu sehen. Darüber sind die Platzhalter, welche die Wortschiffe repräsentieren, zu sehen. Wird ein Wort gefunden, verschwindet auch das entsprechende Schiff. Die linke Seite ist die Fläche, auf der die spielende Person die Wörter finden muss. Auf der rechten Seite sucht der Computer abwechselnd nach dem Spieler nach den versteckten Wörtern.

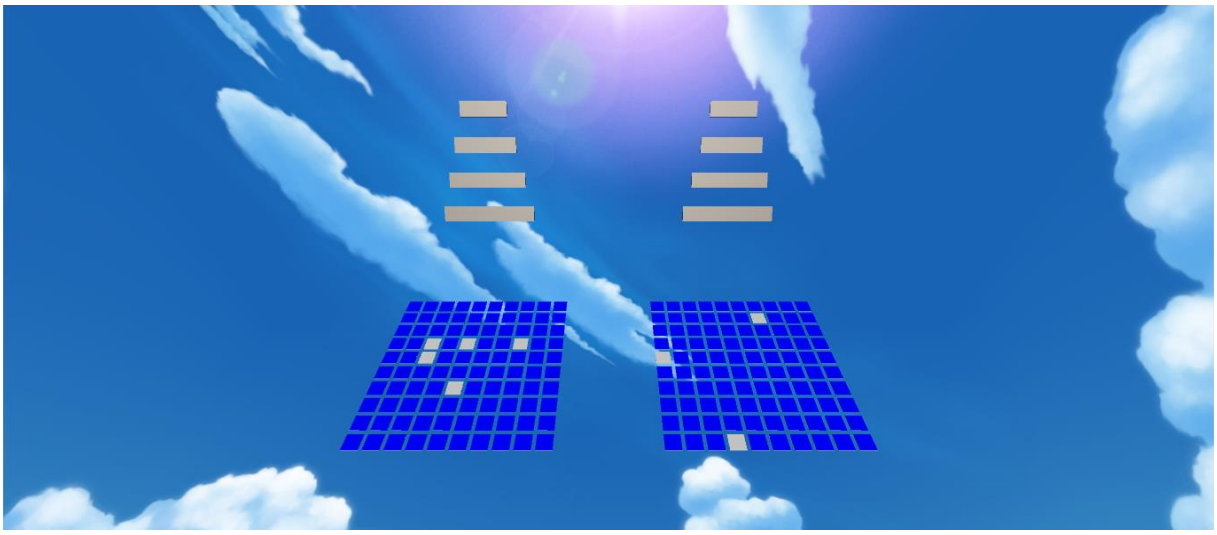


Abbildung 4-55: Spielansicht des Schiffeversenken

Das Kind gewinnt, wenn es alle Wörter vor dem Computer findet. Hat aber der Computer zuvor alle Wörter aufgedeckt verliert die spielende Person das Spiel.

Abbildung 4-56: Voransicht des Buchstaben einsammeln

Startet man den Level sieht man den Bildschirm, welcher in Abbildung 4-56 zu sehen ist. Die spielende Person hat beliebig lange Zeit, das in rot geschrieben Wort zu lesen und sich zu merken. Sobald das Kind bereit ist den Level zu starten, muss es die Enter-Taste drücken.

Abbildung 4-57: Spielansicht des Buchstaben einsammeln

Nach dem Drücken der Eingabetaste muss man dann die Buchstaben in der richtigen Reihenfolge einsammlen. Alle im Wort workommenden Buchstaben werden zufällig auf der Fläche um den Hasen herum aufgeteilt. Der Hase kann mit den Tasten: „w“, „s“, „d“ und „a“ oder mit den Pfeiltasten gesteuert werden. Das Spiel ist gewonnen, wenn alle Buchhstaben eingesammelt wurden und somit das Wort vollständig ist. Wenn man mit dem Hasen in einen Buchstaben fährt, welcher als nächstes ins unvolstädnige Wort gehört, wird dieser zerstört und ins Wort aufgenommen. In Abbildung 4-57 kann man das unvolständige Wort in weißer Farbe erlennen. Die Buchstaben , welche noch eingesammelt werden müssen, sieht man vor dem Hasen ebenfalls in dieser Abbildugn zu sehen.

5 Zusammenfassung

ABC-Darian ist ein Computerspiel für legasthene Kinder. Es ist als Anwendung für Windows, Linux und Mac OS, aber auch als App für mobile Geräte (Android) gedacht.

Entwickelt wurde das gesamte Projekt in Unity3D. Programmiert wurde in der Scriptsprache C# und die Programmierumgebung war MonoDevelop. Modelliert wurden alle benötigten Ob,jekte mit dem Programm Blender. Um die Kinect verwenden zu können, um eine Gestensteuerung gewährleisten zu können, wurde das Microsoft Kinect SDK V1.8 und das zugehörige Toolkit heruntergeladen und verwendet.

Das gesamte Spiel ABC-Darian wurde in mehrere kleine Levels aufgeteilt. Jedes von ihnen stellt ein Minispiel dar und wurde als eigene Scene entwickelt. Eine Levelauswahl und ein Hauptmenü sollten für das Auswählen und das Tätigen von Einstellungen verantwortlich sein. Man kann von jedem Level aus zum Startbildschirm zurückkehren. Da der spielenden Person nicht immer klar ist, wie das Minispiel gespielt wird, gibt es eine Erklärung vor jedem Level. Einerseits gibt es eine schriftliche und bildhafte Erklärung, nachdem ein Level gestartet wurde, welche zuerst gezeigt wird. Zusätzlich kann bei Belieben auch eine auditive Erklärung des Minispiels erfolgen. Im Spiel selbst wird nebenbei eine Melodie abgespielt, um die Levels unterhaltender zu gestalten. Im Hauptmenü kann die Musik aus- oder eingestellt werden.

Beim Starten des Spiels kann man seinen Namen eingeben. Bei vielen Levels gibt es eine Highscore-Liste mit den Namen und Punkten aller SpielerInnen. Nach dem Spielen eines Levels wird ein neuer Eintrag mit dem eingegeben Namen angelegt Somit kann der/die AnwenderIn sich selbst oder andere Spieler in der Punkteanzahl überbieten, was die Minispiele spannender macht.

Das ganze Spiel ist grundsätzlich ohne Kinect möglich. Ist aber eine Kinect verfügbar, so kann in einigen Level, anstatt der herkömmlichen Steuerung das Spiel mit interaktiven Gestensteuerung gespielt werden.

Die Kinect sollte für die Gestensteuerung eingesetzt werden. Doch zu Beginn stellte es ein Problem dar, einen Weg zu finden, sie mit Unity3D zu verwenden. Nach Internetrecherchen und Tipps vom Betreuer wurde das System dann aber gelöst und es wurde eine gute Lösung gefunden. Da das Spiel von zwei Personen entwickelt wurde, hatte man es in Dyskalkulie-Level und Legasthenie-Level aufgeteilt. Eine Problemstellung galt es zu lösen als beide Teile des Spieles, welche beide auf verschiedenen Laptops entwickelt wurden, zu einem großen Programm zusammengefügt werden sollten. Es musste eine Lösung gefunden werden um einen der Programmteile zu exportieren und beim anderen Projektteil hinzugefügt werden. Auch hier half Recherche und Nachfragen bei Mitschülern und beim Projektbetreuer. Am Ende konnte alles problemlos zusammengefügt werden.

Sollte das Projekt jemals weitergeführt werden, wird es am besten um weitere Minispiele erweitert. Es enthält bereits schon eine Vielfalt an Levels, aber für ein richtiges Spiel können es noch mehr sein. Wird das Spiel erweitert, muss darauf geachtet werden das Design beizubehalten. Das Projekt ABC-Darian wurde aber so konzipiert, dass es leicht erweiterbar ist. Dazu muss lediglich eine neue Szene erstellt werden und in der Levelauswahl hinzugefügt werden.

6 Literatur und/oder Quellenverzeichnis

6.1 Allgemeines

[EXP] “3D Formats” (Tutorial), Unity Technologies, http://docs.unity3d.com/Manual/3D-formats.html, 2015

[AFS 01] Dyspedia, “Die AFS-Methode”, http://www.afs-methode.com/index.php/Hauptseite, 27. August 2011

[UNI 14] Unity3D Tutorials, „WHAT IS UNITY?“, http://unity3d.com/pages/what-is-unity, 06 Februar 2015

[STAT 15] Statistics Austria, “Legasthenie in Österreich”, http://www.dyslexiaserver.com/statistik/index.php?action=K%E4rnten, Stand 2015

6.2 Teammitglied: Sereinig

[DYS 01] Bundesverband Legasthenie und Dyskalkulie e. V.: Alena Vanino, „Was ist Dyskalkulie?”, http://www.bvl-legasthenie.de/dyskalkulie.html, 14.09.2014

[DYS 02] Kinder.de – dein Familienportal, Olga Czajowska: „Lernschwächen – Welche gibt es?“, http://www.kinder.de/ratgeber/schule-lernen/lernprobleme/artikel/lernschwaechen-welche-gibt-es.html, 2014

[DYS 03] Bundesverband Legasthenie und Dyskalkulie e. V., Alena Vanino: „Dyskalkulie“, http://www.bvl-legasthenie.de/dyskalkulie/symptomatik.html, 14.09.2014

[DYS 04] Zeit Online: Schule, Petra Hemmelmann: Fehlkalkulation, http://www.zeit.de/2011/29/C-Dyskalkulie/seite-2, 15. Juli 2011

[DYS 05] KIP: Kinderpsychologische Praxis, Mag. Birgit Brenner: „Was kann man gegen eine Rechenstörung tun?“, http://www.kip-kinderpsychologie.at/album1_007.html, 23.Oktober 2014

[BLEN 01]„Blender: Ihr Einstieg in die professionelle 3D-Grafik und Animation“, Heiko Ihde, 01. Oktober 2011

[BLEN 02] Pixeltrain animations, Helge Maus: „Blender 2.6“, http://www.helgemaus.de/tag/videotraining/page/3/, 13.Dezember 2014

[BLEN 03] Blender Tutorial, „Objektmodus“, http://wiki.blender.org/index.php/Doc:DE/2.6/Manual/Modeling/Objects, 09.Oktober 2014

[BLEN 04] „Blender: das Handbuch“, Hanns Ch. Heinrich, 6. Dezember 2014

[KEY 01] Nils Wolf: „Wichtige Befehle und Tastenkombinationen von Blender“, http://wolf-nils.de/index.php/computer/38-blender/64-wichtige-befehle-und-tastenkombinationen, 21. Juni 2014

[FIO 01] Home&Learn, Home&Learn Community: “Write to a text file in C#”, http://www.homeandlearn.co.uk/csharp/csharp_s11p3.html, 03.April 2015

[PP 01] Unity Documentation, Unity Technologies, „PlayerPrefs“, http://docs.unity3d.com/ScriptReference/PlayerPrefs.html, 20.April 2015

6.3 Teammitglied: Zammernig

[WARN 02] Warnke et al, “Leitfaden für die Praxis“, Hogrefe-Verlag, Götting, 2002

[EOEa 95] Dr. Astrid Kopp-Duller, „Pädagogische Definition von Dr. Astrid Kopp-Duller“, http://www.legasthenie.at/EOEDLflyer.pdf, 1995

[EOEb 95] Erster Österreichischer Dachverband Legasthenie, „EOEDL-Flyer“, http://www.legasthenie.at/EOEDLflyer.pdf, 1995

[KIP 13] „Was kann man gegen eine Lese-Rechtschreibstörung tun?“, Kinderpsychologie Praxis, http://www.kip-kinderpsychologie.at/album1_004.htm, 24.1.2014

[KIJ 14] Dr. Hannelore Koch, Mag. Lisa Ehrenhöfler und Mag. Neeltje Obergfell „Lese-Rechtschreibstörung/Legasthenie“, http://www.kijupraxis.at/KiJuPraxis/Diagnostik%26_Behandlung/Eintrage/2010/7/20_Lese-Rechtschreibstorung__Legasthenie.html, 29.12.2014

[VGN 10] Erica Naone, „Vom GUI zum NUI“, http://www.heise.de/tr/artikel/Vom-GUI-zum-NUI-1107776.html, 14.10.2010

[WIT 14] „Windows Touch“, http://windows.microsoft.com/de-de/windows7/products/features/touch, 15.10.2014

[KFW 14] „Kinect for Windows“, http://www.microsoft.com/en-us/Kinectforwindows/, 8.10.2014

[NUI 13] Wolfgang Henseler „Natural User Interfaces – Die Kunst, Nutzung intuitiv zu gestalten“, https://webmagazin.de/allgemein/natural-user-interfaces-die-kunst-nutzung-intuitiv-zu-gestalten-4095000, 25.6.2013

[SDK 14] Kinect for Windows SDK v1.8”, http://www.microsoft.com/en-us/download/details.aspx?id=40278, 25.10.2014

[GDTP 04] Rouse Richard 3, „Game Design: Theory & Practice“, Wordware Pub Co, 2. Auflage, 1.9.2004

[BSP 15] „Beispiel 1: Fliegende Tiere“, http://www.legastheniepraxis.at/legasthenie/original_beispiel1.html, 16.10.2015

[OSI 15] “OSIT – natürliches Handlungsschema“, https://webmagazin.de/allgemein/natural-user-interfaces-die-kunst-nutzung-intuitiv-zu-gestalten-4095000, 13.12.2015

[KIN 15] „Kinect 360“, http://bilder.pcwelt.de/1952261_original.jpg, 1.2.2015

[BLS 15] „Blockschaltbild Kinect“, https://www.ifixit.com/Guide/Image/meta/aT2uupuVXHnJKTsU, 3.1.2015

7 Anhang (Dokumente und Protokolle)

7.1 Projektdokumentation

7.1.1 Pflichtenheft

Kandidat / Kandidatin	Vanessa Sereinig
Thema	Programmdesign
Aufgabenstellung	
Pflichtenheft	▪ Erstellen des Pflichtenheftes gemeinsam mit dem Projektpartner
Auswahl und Kennenlernen der Entwicklungsumgebung	▪ Einlesen in die Programme „Unity 3D“ und „Blender“. Abwägen der nutzbaren Techniken zum Erstellen der Spiele unter Berücksichtigung der recherchierten Behandlungsmöglichkeiten.
Erstellung User Interface	▪ Gestaltung des GUI in Unity 3D. ▪ Geforderte User Interfaces: Einstellungen und Optionen, Levelauswahl, Anzeige über den Lernfortschritt, Startmenü (Charakterauswahl,…), usw. ▪ Methode der Spielstandsspeicherung auswählen.
Levelentwicklung	▪ Gestaltung der 2D/3D-Welt, in der sich die Third Person bewegt. ▪ Auswahl der optimalsten Programmiermethode zum Erstellen und Einbinden der verschiedenen Blender-Objekte.
Blender Objekterstellung	▪ Zusammen mit dem Projektpartner die erforderlichen Objekte (Spielcharaktere, Gegenstände, Umgebung) besprechen und diese eigenständig in Blender kreieren.
Gemeinsame Levelgestaltung	▪ Absprache des generellen Levelaufbaues gemeinsam mit dem Projektpartner, um eine individuelle Einzelarbeit mit anschließender Kombination zu erreichen.
Recherche zu Dyskalkulie	▪ Anstellung diverser Recherchen zu dem Thema Dyskalkulie. Die Nachforschung sollte beinhalten: o Behandlungsmöglichkeiten o auffälligste und häufigste Ausprägungen v. Dyskalkulie o beliebte Lernspiele o Optimaler Umfang und Schwierigkeitsgrad der einzubindenen Rechenschritte
Dokumentation	▪ Dokumentation lt. Anforderung der HTL Mössingerstraße

Kandidat / Kandidatin	Andreas Zammernig
Thema	Einbindung der Kinect
Aufgabenstellung	
Pflichtenheft	▪ Erstellen des Pflichtenheftes gemeinsam mit der Projektpartnerin
Einüben in die Entwicklungsumgebung	▪ Einlesen in „Unity 3D", sowie Kennenlernen der Integrationsmöglichkeiten der Kinect. ▪ Abschätzung und Besprechung des Spielkonzeptes unter Berücksichtigung der Recherchen zu den sinnvollen Lernmethoden zusammen mit der Projektpartnerin.
Einbindung Kinect	▪ Integration der Kinect in die Entwicklungsumgebung von Unity 3D mit der KINECT-SDK. ▪ Sicherstellung der Kommunikation zwischen Kinect und Anwendung. ▪ Erstellung diverser Testlevel zur Verifizierung. ▪ Entwicklung zusätzlicher Aufgaben, welche nur auf der Kinect basieren und dem Nutzer eine Avatarsteuerung in der Spieleumgebung ermöglichen.
Gemeinsame Levelgestaltung	▪ Absprache des generellen Levelaufbaues gemeinsam mit der Projektpartnerin, um eine individuelle Einzelarbeit mit anschließender Kombination zu erreichen.
Recherchen zu Legasthenie	▪ Durchführung diverser Recherchen zu dem Thema Legasthenie. Die Nachforschung sollte beinhalten: o Behandlungsmöglichkeiten o Häufig anfallende Rechtschreib- und Leseschwächen o Beliebteste Lernspiele mit größtem Lernerfolg
Dokumentation	▪ Dokumentation lt. Anforderung der HTL Mössingerstraße

7.1.2 Systemstruktur

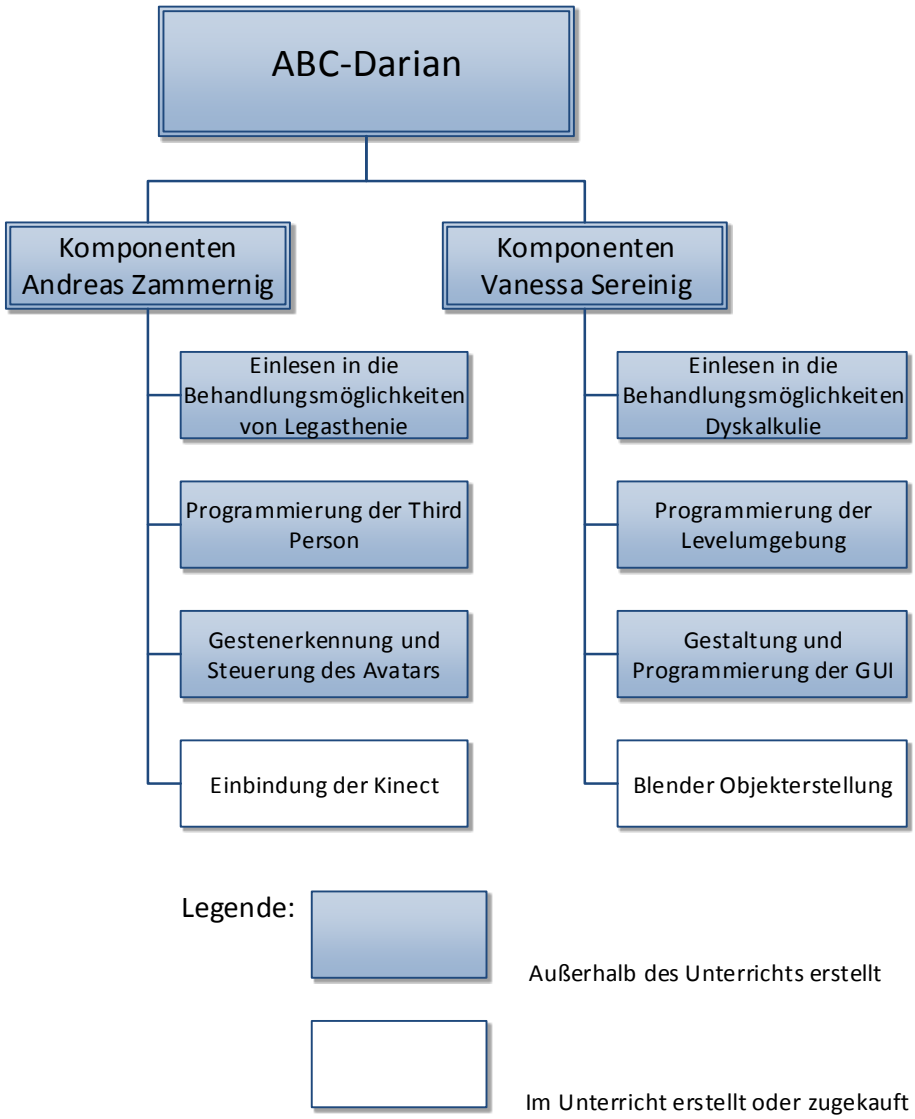


Abbildung 7-1: Systemstruktur

7.1.3 Projektstruktur

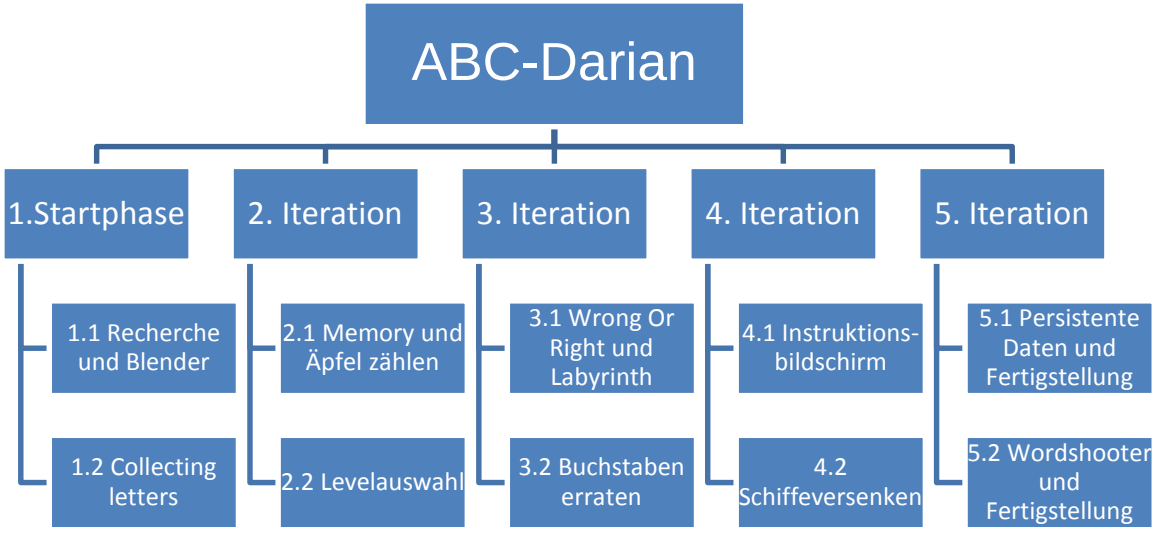


Abbildung 7-2: Projektstruktur

7.1.4 Risikoabschätzung

Bezeichnung	Wahrscheinlichkeit	Beschreibung
Programmierung der Kinect	sehr niedrig	Es muss sichergestellt werden mit der Kinect zu arbeiten. Hierfür müssen Treiber und SDK installiert werden. Es kann erst zu entwickeln begonnen werden, wenn eine Kommunikation mit der Kinect hergestellt werden kann.
Zusammenfügen der Projektteile	niedrig	Am Ende muss das Projekt zusammengefügt werden. Da beide Personen ihren eigenen Teil entwickelt haben, kann es hier zu Problemen kommen. Problemstellungen können verschiedene Versionen, Kompatibilitätsprobleme oder ähnliches sein.
Kompatibilität der Kinect mit der Hardware	mittel	Im Gegensatz zum eigentlichen Spiel ist die Software, die für die Kinect benötigt wird, weniger kompatibel. Die notwenige Microsoft Kinect SDK und Kinect-Treiber kann nur auf gewissen Windows-Betriebssystemen installiert werden.

7.1.5 Terminplan

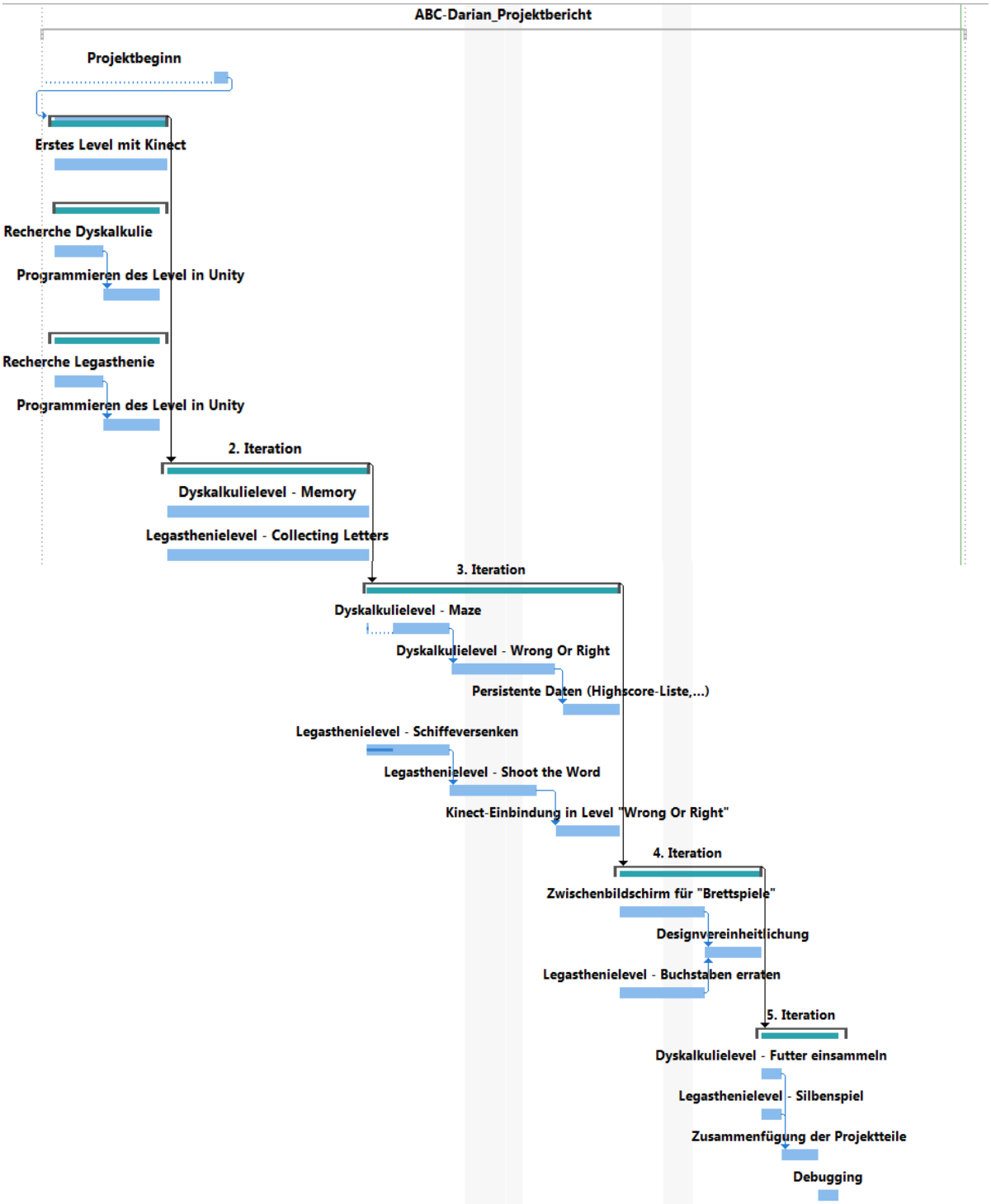


Abbildung 7-3: Terminplan in MS Project

7.2 Arbeitszeitnachweise

Datum	Start	Ende	Fortschritt	Dauer [h]	Stunden gesamt
19.09.2014	16:15	18:45	Beginn theoretische Arbeit (Seiten 1-3)	2,50	
29.09.2014	17:00	19:00	Fortsetzung theortische Arbeit(Seiten 4-6)	2,00	159,3
05.10.2014	19:00	21:00	Diverse Blender Tutorials (Videos)	2,00	
06.10.2014	18:00	20:30	Fortsetzung theortische Arbeit; Gestaltung StartMenü	2,50	
07.10.2014	19:00	23:00	Fortsetzung theortische Arbeit (bis Seite 9); Gestaltung Scene02	4,00	
16.10.2014	18:00	21:00	Beginn Level "CountCubes", Entwicklung Algorithmus für 2D	3,00	
20.10.2014	20:00	22:00	Verfeinerung von "EvaluateBiggest", Äpfel hinzugefügt	2,00	
24.10.2014	18:00	21:00	Level "CountCubes", Drehversuche durch Benutzer, Start 3D-Algorithmer	2,00	
31.10.2014	15:00	18:00	Fortsetzung theoretische Arbeit, Blender Tutorials	3,00	
04.11.2014	19:00	20:30	Cartoon-Gestaltung von "EvaluateBiggest"	1,50	
07.11.2014	17:00	21:00	Level "Memory" - Start	4,00	
13.11.2014	16:00	20:00	Level "Memory" - Fortsetzung	4,00	
14.11.2014	18:00	21:00	Gestaltung der Memorykarten (Texturen), Erweiterung "Memory"	3,00	
16.11.2014	21:00	23:00	Zufallsalgorithmen der Memorykarten für "Memory"	2,00	
19.11.2014	16:40	18:00	Vorbereitung für die Wettbewerbsanmeldungen rtner	1,30	
20.11.2014	17:00	19:30	Fehlerbehebungsversuche, Umschreiben der Algorithmen	2,50	
24.11.2014	16:00	22:00	Fertigstellung "Memory", Fehlerbehebung, Fertigstellung der Texturen	6,00	
25.11.2014	19:00	21:00	Project Management, Fertigstellung der Aufgaben bis II.Iteration	2,00	
26.11.2014	12:30	18:00	Vorbereitung für die 2. Iteration, Verschönerung der Level	5,00	
26.11.2014	19:00	23:00	Fortsetzung Theoretische Arbeit, Bearbeitung der Gesamtdokumenatior	4,00	
05.12.2014	17:00	19:00	Texteingabe für "CountCubes", Bearbeitung Design	2,00	
18.12.2014	16:00	18:00	Level "Maze" - Start, Überlegung, Recherche	2,00	
22.12.2014	15:00	19:00	Design für das Labyrinth in Blender, Gestaltung und Modellierung	4,00	
26.12.2014	09:00	11:00	Level "Wrong Or Right" - Start (Ideensammeln, Texturensuche)	2,00	
27.12.2014	11:00	14:00	Level "Maze" - Beginn des Scripts, Algorithmus für Zahlenberechnung	3,00	
04.01.2015	13:00	16:30	Level "Maze" - Dynamische Zahlen(Objekt-)erzeugung im Labyrinth	3,50	
07.01.2015	14:00	17:30	Fortsetzung der Doku- Dokumentation der eigenständigen Arbeit	3,50	
11.01.2015	10:00	12:00	Level "Maze" - Fertigstellung, Einzelheiten hinzugefügt	2,00	
14.01.2015	18:00	19:00	Tutorial für persistente Daten (Video)	1,00	
18.01.2015	14:00	16:30	Level "Wrong Or Right" - Fortsetzung	2,50	
23.01.2015	13:00	15:00	Project Management, Fertigstellung der Aufgaben bis III.Iteration	2,00	
24.01.2015	17:00	19:30	Fertigstellung von "Wrong Or Right", Erstellung zweiter Version	2,50	
25.01.2015	16:00	19:30	Erstellen einer Highscore Liste, Beschäftigung mit Persistenten Daten	3,50	
26.01.2015	20:00	22:00	Fortsetzung Doku	2,00	
27.01.2015	10:00	11:30	Vorbereitung der Präsentation sowie Powerpoint	1,50	
27.01.2015	19:00	00:00	Level "CollectNuts" - Start, Design der Welt Modellierung der Umgebung	5,00	
20.02.2015	15:00	18:00	Skybox Erstellung	3,00	
25.02.2015	17:00	20:00	Beginn Vereinheitlichung des Spiele-Designs	3,00	
27.02.2015	15:00	18:00	Beginn des Demo Levels (Bunny läuft auf Tisch zu	3,00	
28.02.2015	20:00	22:30	Dokumentation des Projektberichts für JI	2,50	
01.03.2015	17:00	19:30	Adaption von Maze; Erstellung des Instructionbildschirms	2,50	
02.03.2015	19:00	21:30	Dokumentation (Fertigstellung) des Projektberichts für JI	2,50	
03.03.2015	18:00	22:00	Fertigstellung Instruktionsbildschirm, Fertigstellung bisheriger Level	4,00	
05.03.2015	20:00	23:00	Dokumentation	3,00	
08.03.2015	16:00	17:30	Dokumentation (Ergebnisse)	1,50	
11.03.2015	16:00	19:30	Fertigstellung des DemoLevels mit Einbindung von Animation	3,50	
12.03.2015	17:00	21:30	Doku-Fortsetzung (Überarbeitung der bisherigen Realisierung)	4,50	
13.03.2015	18:00	21:30	Anmeldung zu weiteren Wettbewerbung gemeinsam mit Projektpartner	3,50	
15.03.2015	09:00	13:00	Anfertigung des Businessplans für die Gesamtdokumentation	4,00	
20.03.2015	18:00	20:00	Fertigstellung der meisten "Letters with Background" in Blender	2,00	
23.03.2015	17:00	20:00	Beginn der Levelprogrammierung von "CollectNuts"	3,00	
26.03.2015	14:00	17:00	Beginn der Änderung des Instruktionbildschirm - eigene Szene designen	3,00	
27.03.2015	15:00	18:00	Debugging	3,00	
30.03.2015	10:00	12:00	Fertigstellung Instructions-Scene; hilfreiche Kommentierung	2,00	
02.04.2015	14:00	17:00	Fertigstellung "CollectNuts", Einbindung Audio-Dateien	3,00	
05.04.2015	15:00	17:30	Fertigstellung HighScore-Lists, Hinzufügen von Spielerprofilen	2,50	

Datum	Start	Ende	Fortschritt	Dauer [h]	Stunden gesamt
28.08.2014	14:00	18:00	Strukturierung und Themenauswahl der DA	4,00	161,0
29.08.2014	09:00	12:00	Beginn der theoretischen DA	3,00	
30.08.2014	15:00	18:30	Recherche Legasthenie - Allgemeines	3,50	
13.09.2014	13:00	17:00	Microsoft SDK Installation	4,00	
14.09.2014	09:00	12:00	Treiber Installation und Test der Kinect	3,00	
20.09.2014	13:00	16:00	Test der Kinect in Unity3D	3,00	
21.09.2014	10:00	12:00	Implementierung einer ersten Steuerung mittels Gestenerkennung	2,00	
27.09.2014	14:00	17:00	Recherche Legasthenie - Definition	3,00	
05.10.2014	15:00	19:00	Recherche Legasthenie - Erscheinungsbild	4,00	
17.10.2014	16:00	20:00	Ausprogrammieren der Gestenerkennung	4,00	
18.10.2014	17:00	20:00	Implementierung der Levelauswahl - Grundsatz	3,00	
19.10.2014	09:00	13:00	Erstellen der Microsoft Project Datei	4,00	
25.10.2014	09:00	11:00	Implementierung der Levelauswahl - Erweiterungen	2,00	
26.10.2014	09:00	13:00	Level Buchstaben einsammeln - Grundsatz	4,00	
01.11.2014	16:00	20:00	Level Buchstaben einsammeln - Erweiterung	4,00	
08.11.2014	14:00	16:00	Recherche NUI - Definition	4,00	
16.11.2014	09:00	12:00	Level - Buchstaben erraten - Grundsatz	3,00	
22.11.2014	17:00	20:00	Recherche Kinect - Allgemein	3,00	
23.11.2014	08:00	12:00	Recherche OSIT	4,00	
26.11.2014	13:05	17:05	Level - Buchstaben erraten - Erweiterungen	4,00	
06.12.2014	09:00	13:00	Level Schifferversenken - Grundgerüst	4,00	
13.12.2014	15:00	19:00	Level Schifferversenken - Texturen einbinden	4,00	
22.12.2014	14:00	18:00	Level Schifferversenken - Gegner spielt mit	4,00	
23.12.2014	09:00	12:00	Recherche Unity	3,00	
26.12.2015	13:30	18:00	Level ShootTheWord - Grundgerüst	4,50	
27.12.2015	09:00	12:00	Level ShootTheWord - Punktesystem	3,00	
28.12.2015	14:00	17:00	Level ShootTheWord - individuelle Objekte zum Abschießen	3,00	
10.01.2015	08:30	12:00	Kinectintegration in das Level - RightOrWrong	3,50	
17.01.2015	14:00	17:30	Level Schifferversenken - Fehlerbehebung	3,50	
24.01.2015	14:00	17:30	Diplomarbeit - Website	3,50	
27.01.2015	13:00	17:30	Vorbereitung 3. Iteration (Englisch)	4,50	
01.02.2015	14:00	17:00	Level Collecting Letters - Design überarbeitet	3,00	
07.02.2015	13:00	17:30	Level ShootTheWord - neues Spieldesign und Darstellung	4,50	
15.02.2015	13:30	17:00	Level Levelauswahl - Design an andere Level angepasst	3,50	
21.02.2015	14:00	17:00	Level ShootTheWord - neuer Algorithmus zur Punkterfassung	3,00	
28.02.2015	13:00	17:00	Recherche Legasthenie - Behandlung	4,00	
07.03.2015	09:00	11:00	Level Levelauswahl - Mit Animationen versehen	3,00	
13.03.2015	08:00	10:00	Level Schifferversenken - Neues Design und Optimierung	3,00	
14.03.2015	15:00	19:00	Einfügen des Hasenmodells und anderer Umgebungsobjekte	4,00	
19.03.2015	08:00	13:00	Recherche Gamedesign	5,00	
21.03.2015	14:00	17:00	Recherche Leveldesign	3,00	
28.03.2015	13:00	18:00	Erarbeiten eines gemeinsamen Design	5,00	
01.04.2015	08:00	12:00	Übernehmen des eines gemeinsamen Design auf alle Level	4,00	
03.04.2015	15:00	18:00	Debuggen - Buchstaben einsammeln und Buchstaben erraten	3,00	
07.04.2015	08:00	12:00	Debuggen - Levelauswahl und ShootTheWord und Schiffeversenken	4,00	

Tabelle 7-2: Arbeitszeitennachweis Zammernig

7.3 Technische Dokumentation

7.3.1 Unity3D

Das gesamte Projekt wurde in Unity3D entwickelt.

7.3.1.1 Links

Bezeichnung	Link
Download	http://unity3d.com/get-unity
Tutorials	http://unity3d.com/learn/tutorials/modules
Dokumentation	http://docs.unity3d.com/Manual/index.html
Scripting API	http://docs.unity3d.com/ScriptReference/

Tabelle 7-3: Links von Unity

7.3.1.2 Versionen

Unity3D ist in 2 Versionen erhältlich. Eine kostenfreie Version und eine kostenpflichtige Pro-Version. Für die Entwicklung von ABC-Darian wurde die gratis Version verwendet. Eine Auflistung der jeweiligen Features kann in Abbildung 7-5 eingesehen werden.

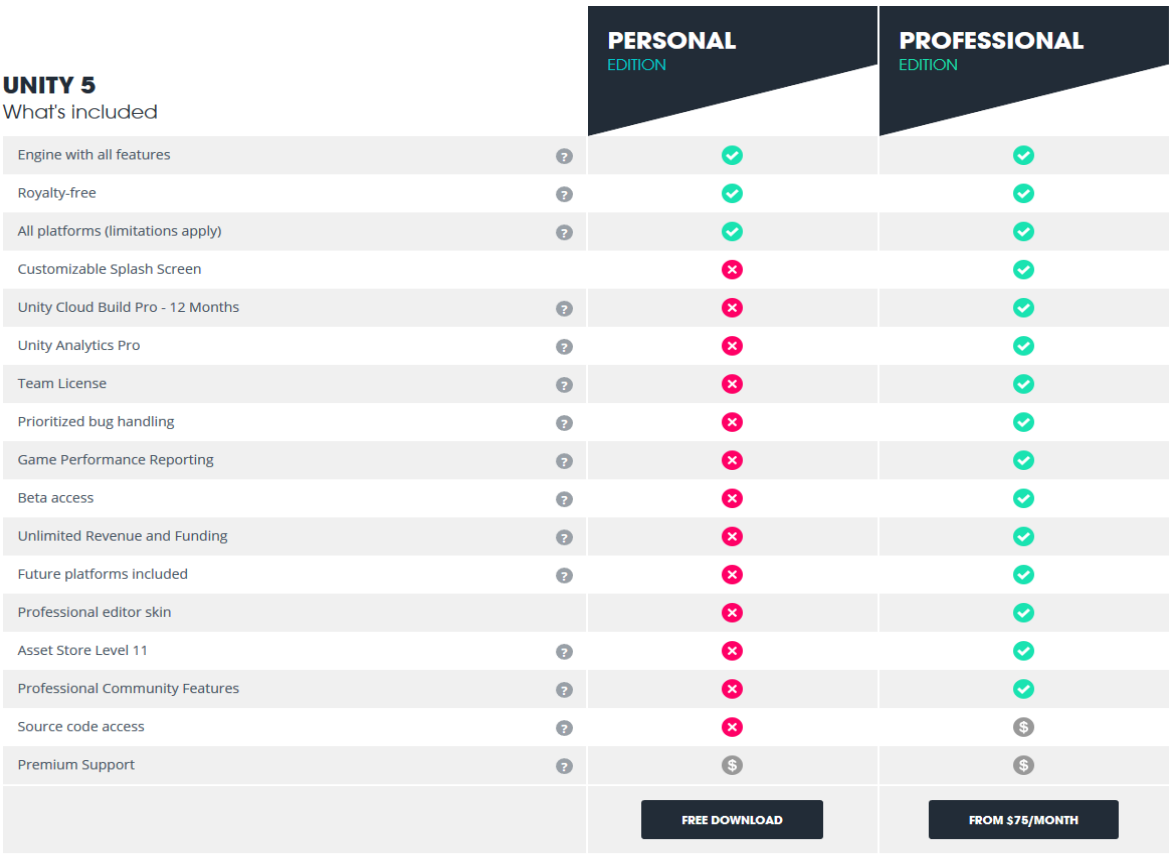


Abbildung 7-5: Versionen von Unity3D

7.3.1.3 Systemvoraussetzungen:

OS

- Windows XP SP2+, 7 SP1+, 8
- Mac OS X 10.8+

GPU

- Grafikkarte mit DX9-Unterstüzung(Shader Model 2.0)

7.3.2 Blender

Zur Modellierung der einzelnen Objekte wurde das kostenfreie Programm Blender verwendet.

7.3.2.1 Links

Tabelle 7-4: Links von Blender

Bezeichnung	Link
Download	https://www.blender.org/download/
Tutorials	https://www.blender.org/support/tutorials/
Anleitung	https://www.blender.org/manual/

7.3.2.2 Systemvoraussetzungen

7.3.2.2.1 Mindestanforderungen

- 32-bit dual core 2Ghz CPU mit SSE2 support
- 2 GB RAM
- 24 Bits 1280×768 Display
- Maus oder Trackpad
- OpenGL-kompatible Grafikkarte mit 256 MB RAM

7.3.2.2.2 Empfohlen

- 64-bit quad core CPU
- 8 GB RAM
- Full HD-Display mit 24 bit-Farben
- Drei-Tasten-Maus mit Mausrad
- OpenGL-kompatible Grafikkarte mit 1 GB RAM

7.3.2.2.3 Optimal

- 64-bit eight core CPU
- 16 GB RAM
- zwei Full HD-Displays mit 24-bit Farben
- Drei-Tasten-Maus mit Mausrad und Grafiktablet
- Dual OpenGL-kompatible Grafikkarten von Qualitätsmarken mit 3 GB RAM

7.3.3 Kinect

7.3.3.1 SDK Version 1.8

Sowohl die Kinect SDK Version 1.8 als auch das zugehörige Toolkit sind gratis erhältlich. Es kann gemeinsam auf der offiziellen Seite von Microsoft heruntergeladen werden. Für Untiy3D ist auch ein entsprechendes Tutorial im Asset Store von Unity3D verfügbar.

7.3.3.1.1 Links

Tabelle 7-5: Links für Kinect SDK 1.8

Bezeichnung	Link
Download	http://www.microsoft.com/en-us/download/details.aspx?id=40278
Bekannte Probleme	https://msdn.microsoft.com/en-us/library/dn435682.aspx
Dokumentation	https://msdn.microsoft.com/en-us/library/jj663803.aspx#SDK_1pt8
Tutorial	https://www.assetstore.unity3d.com/en/#!/content/7747

7.3.3.1.2 Unterstütze Betriebssysteme:

- Windows 7
- Windows 8
- Windows 8.1
- Windows Embedded Standard 7

7.3.3.1.3 Hardwarevoraussetzungen:

- 32-Bit- (x86) oder 64-Bit- (x64) Prozessor
- Dual-core 2.66-GHz oder ein schnellerer Prozessor
- Dedicated USB 2.0 Bus
- 2 GB RAM
- Eine Microsoft Kinect, welche für Windows geeignet ist

7.3.3.1.4 Softwarevoraussetzungen:

- Visual Studio 2010 or Visual Studio 2012 Express

- .NET Framework 4 (installiert mit Visual Studio 2010), oder.NET Framework 4.5 (installiert mit Visual Studio 2012)

7.3.3.2 Kinect SDK Developer Toolkit Version 1.8

7.3.3.2.1 Link

Tabelle 7-6: Link für Kinect Developer Toolkit SDK 1.8

7.3.3.2.2 Unterstütze Betriebssysteme:

Bezeichnung	Link
Download	http://go.microsoft.com/fwlink/?LinkID=323589

- Windows 7
- Windows 8
- Windows 8.1
- Windows Embedded Standard 7

7.3.3.2.3 Hardwarevoraussetzungen:

- 32-Bit- (x86) oder 64-Bit- (x64) Prozessor
- Dual-core 2.66-GHz oder ein schnellerer Prozessor
- Dedicated USB 2.0 Bus
- 2 GB RAM
- Eine Microsoft Kinect, welche für Windows geeignet ist

7.3.3.2.4 Softwarevoraussetzungen:

- Kinect SDK v1.8 für Windows

7.4 Arbeitspakete

ARBEITSPAKETE - ABHÄNGIGKEITEN		Schuljahr: 2014/15
PROJEKT:	ABC-Darian	Abteilung: Elektronik und Technische Informatik
PROJEKTLEITER:	Vanessa Sereinig	Datum: 15.04.2015

Code	Projektphase / Arbeitspaket	Beschreibung	Abhängigkeiten
1	Phase 1	Startphase	
1.1	Arbeitspaket 1.1.1	Recherche und Blender	
1.2	Arbeitspaket 1.2.1	Collecting Letters	
2	Phase 2	2. Iteration	
2.1	Arbeitspaket 2.1.1	„Memory“ und „Äpfel zählen“	
2.2	Arbeitspaket 2.2.1	Levelauswahl	
3	Phase 3	3. Iteration	
3.1	Arbeitspaket 3.1.1	„Wrong Or Right“ und “Labyrinth”	
3.2	Arbeitspaket 3.2.1	Buchstaben erraten	
4	Phase 4	4. Iteration	
4.1	Arbeitspaket 4.1.1	Instruktionsbildschirm	
4.2	Arbeitspaket 4.2.1	Schiffeversenken	
5	Phase 5	5. Iteration	
5.1	Arbeitspaket 5.1.1	Persistente Daten und Fertigstellung	
5.2	Arbeitspaket 5.2.1	Wordshooter und Fertigstellung	

ARBEITSPAKET Nr. 1.1.1	**Schuljahr: 2014/15**
PROJEKT: **ABC-Darian**	Abteilung: Elektronik und Technische Informatik
PROJEKTPHASE: Startphase **ARBEITSPAKET:** Recherche und Blender	Datum: 08.10.2014 Bearbeiter: Sereinig e-mail: vanessa.sereinig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel
Es sollten zu Beginn erste Nachforschungen zum Thema Dyskalkulie getätigt werden. Besonderes Augenmerk sollte auf die Behandlungsmöglichkeiten und die häufigsten Fehler gelegt werden. Nebenbei sollte ein erster Level in Unity3D begonnen werden, um eine Einführung in die Entwicklungsumgebung und die Levelgestaltung zu erhalten. Auch in Blender soll sich eingelesen werden.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Die Nachforschungen zum Thema Lernschwächen gab einige Ideenanreize für das Design von Levels. Das Einlesen in Blender erfolgte weitgehend durch das Ansehen von verschiedenen Video-Tutorials. Die ersten gestalteten Objekte sind dreidimensionale Buchstaben und Zahlen, da diese essentiell für das Arbeiten im dreidimensionalen Raum sind. Auch Satzzeichen und ein Mülleimer wurden modelliert. Das ersten gestalteten Szenen in Unity3D sind zum einem das Spiel „Zahlen einsammeln“ und zum anderen das Startmenü. Ersteres wurde mit einer Third Person und den zuvor modellierten, dreidimensionalen Zahlen realisiert. Der Benutzer muss hierbei das Ergebnis einer angezeigten Rechnung einsammeln. Beim Einsammeln einer falschen Zahl kann der Mülleimer verwendet werden. Das Startmenü ist dazu da, um das Spiel zu starten, Einstellungen vorzunehmen und Co.	Die Vorgaben konnten bis zum vereinbarten Zeitpunkt vollständig und zufriedenstellend eingehalten werden. Es traten keine Probleme bei den Ausführungen auf.

ARBEITSPAKET Nr. 2.1.1	**Schuljahr: 2014/15**

PROJEKT:	**ABC-Darian**	Abteilung: Elektronik und Technische Informatik
PROJEKTPHASE:	2. Iteration	Datum: 27.11.2014
ARBEITSPAKET:	Memory und Äpfel zählen	Bearbeiter: Sereinig
		e-mail: vanessa.sereinig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel

Es sollen zwei Level, „Memory“ und „Äpfel zählen“, realisiert werden. Die dafür nötigen Objekte werden in Blender designt, die Texturen für das Memory in Paint.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
• Memory Memory ist wie das herkömmliche Brettspiel, nur dass das Kind anstatt komplett gleicher Bilder die Rechnung mit dem passenden Ergebnis aufdecken muss. Es kann zwischen 8, 16, 24 und 36 Karten gewählt werden. • Äpfel zählen Dieses Spiel fordert das Kind im Sinne vom Mengen verstehen. Es werden zwei Zahlen angezeigt, welche zusätzlich mit Apfelpyramiden optisch dargestellt werden. Es muss nun herausgefunden werden, welcher Wert größer ist. Somit kann das Kind einen Bezug zu Zahlen und den Mengen, die sie repräsentieren, aufbauen. Die Äpfel für das Level wurden in Blender designt und modelliert.	Die Vorgaben konnten bis zum vereinbarten Zeitpunkt vollständig und zufriedenstellend eingehalten werden. Es traten keine Probleme bei den Ausführungen auf.

ARBEITSPAKET Nr. 3.1.1	**Schuljahr: 2014/15**

PROJEKT:	**ABC-Darian**	Abteilung: Elektronik und Technische Informatik
PROJEKTPHASE:	3. Iteration	Datum: 28.01.2015
ARBEITSPAKET:	„Wrong Or Right“ und “Labyrinth”	Bearbeiter: Sereinig
		e-mail: vanessa.sereinig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel

Das Level Buchstaben erraten, sollte als nächstes realisiert werden. Hierbei geht es darum ein Wort zu erraten anhand seines Anfangsbuchstaben. Mit diesem Level sollte auch wieder bei dem Kind das Gefühl für Wörter gegeben werden.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Mit dem Wissen über die Behandlungsmöglichkeiten von Legasthenie wurde das nächste Ldevel entworfen. Ein zufällig ausgewäheltes Wort wird in ein Textfeld geschrieben. Beim Wort selbst ist nur der erste Buchstabe zu sehen. Eine Tastatur die per Script erstellt wird, dient dazu Buchstaben in das Wort einzufüllen. Ein Turm aus mehern Zylindern mit einem Hasen an der Spitze wurde erstellt, um den Level spannender zu machen. Ein Algorithmus stellt sicher, dass bei einem falsch ausgewähelten Buchstabe der Turm kleiner wird und bei einem Buchstaben der in das Wort passt, dieser eingefügt wird.	Die Vorgabe wurde bis zum vereinbarten Datum pünktlich und richtig fertiggestellt. Es traten keine Probleme bei der Durchführung dieser Arbeit auf.

ARBEITSPAKET Nr. 4.1.1	**Schuljahr: 2014/15**

PROJEKT:	**ABC-Darian**	Abteilung: Elektronik und Technische Informatik
PROJEKTPHASE:	4. Iteration	Datum: 04.03.2015
ARBEITSPAKET:	Instruktionsbildschirm	Bearbeiter: Sereinig e-mail: vanessa.sereinig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel

Um dem Kind zu erklären, wie jeder der Minispiele funktioniert, ist ein Intruktionsbildschirm wichtig. Dieser sollte vor dem Start eines jeden Levels angezeigt werden und mit Bildern und Text das folgende Spiel erklären.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Der Instruktionsbildschirm wurde erst als Skript realisiert. Als hierbei Fehler auftraten, vor allem die GUI betreffend, wurde eine ganze eigene Szene dafür designt. In der Szene befindet sich nichts außer einer Skybox und der Kamera, auf der das Skript liegt. Die GUI des Intruktionsbildschirms wurde dynamisch designt. An Hand des nachfolgenden Levels (es wird eine Integer-Variable mit dem Built-Index des Minispiels übergeben) wird ausgewählt, welche Texturen gesetzt und welcher Text angezeigt wird. Hierfür gibt es eigene Variablen, die in einer switch-Anweisung gesetzt werden. Es können entweder ein Text und eine Texture angezeigt werden, oder ein Text und zwei Texturen. Die Texturen sind Bilder, welche die Steuerung erklären bzw. in manchen Fällen ein Beispiel darstellen. Damit das Kind beim Lesen keine Verständnisschwierigkeiten hat, gibt es außerdem die Möglichkeit, sich die Texte vorlesen zu lassen. Hierfür gibt es einen eigenen Button, der ein aufgenommenes Audio-File abspielt.	Die Vorgaben konnten bis zum vereinbarten Zeitpunkt vollständig und zufriedenstellend eingehalten werden. Es traten keine Probleme bei den Ausführungen auf.

ARBEITSPAKET Nr. 5.1.1	**Schuljahr: 2014/15**
PROJEKT: **ABC-Darian**	Abteilung: Elektronik und Technische Informatik

PROJEKTPHASE:	5. Iteration	Datum:	15.04.2015
ARBEITSPAKET:	Persistente Daten und Fertigstellung	Bearbeiter:	Sereinig
		e-mail:	vanessa.sereinig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel
Gemeinsam mit dem Projektpartner soll das Projekt in sich fertiggestellt werden, dies beinhaltet das Zusammenführen aller Szenen und restlichen Assets. Außerdem sollen zuvor noch die persitente Datenspeicherung und das Level „Futter sammeln“ abgeschlossen werden.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Im Spiel „Futter sammeln“ ist es dem Benutzer möglich, sich auf einer Fläche zu bewegen und herumliegendes Futter mit der Third Person einzusammeln. Es wird dabei eine Anzahl vorgegeben, die eingesammelt und in eine Schüssel gelegt werden soll. Wichtig dabei ist, dass der Hase (Third Person) nur maximal vier Futterstücke auf einmal tragen kann. Somit wird dem Kind nicht nur ein prinzipielles Gefühl für das Zählen vermittelt, sondern es wird auch mit dem Thema Addition konfrontiert. Die persistente Datenspeicherung wurde mittels PlayerPrefs realisiert. Diese Klasse macht es einem möglich, Daten wie Namen und Punkteanzahl ganz einfach permanent zu speichern. Somit konnte für jedes Minispiel eine individuelle Highscore-Liste gestaltet werden. Die Highscore-Liste ist eine eigene Szene und wird nach Beendigung eines Levels aufgerufen. Zum Schluss wurde noch da gesamtes Spiel zusammengefügt und auf dasselbe Design gebracht. Es wurde auch sichergestellt, dass es als Gesamtprojekt funktioniert und alle Fehler behoben sind.	Die Vorgaben konnten bis zum vereinbarten Zeitpunkt vollständig und zufriedenstellend eingehalten werden. Es traten keine Probleme bei den Ausführungen auf.

ARBEITSPAKET Nr. 1.2.1	**Schuljahr: 2014/15**
PROJEKT: **ABC-Darian**	Abteilung: Elektronik und Technische Informatik
PROJEKTPHASE: Startphase **ARBEITSPAKET:** Collecting Letters	Datum: 08.10.2014 Bearbeiter: Zammernig e-mail: andreas.zammernig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel

Es sollten zu Beginn erste Nachforschungen zu Legasthenie getätigt werden. Besonderes Augenmerk sollte auf die Behandlungsmöglichkeiten gelegt werden und die häufigsten Fehler. Nebenbei sollte ein erster Level in Unity3D begonnen werden, um eine Einführung in die Entwicklungsumgebung und die Levelgestaltung zu erhalten.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Es wurde mit der Nachforschung zum Thema Legasthenie begonnen. In die Entwicklungsumgebung Unity3D und die Programmierumgebung MonoDevelop sollte sich ebenfalls eingelesen werden. Dabei wurden die effektivsten Behandlungsmöglichkeiten behandelt. Parallel zur Recherche wurde ein Projekt in Unity3D erstellt und begonnen ein erstes Level zu gestalten. Die erste Scene, welche das erste Minispiel darstellte, sollte den Level „Buchstaben einsammeln“. Hierfür wurden die Spielfläche und erste Objekte eingefügt. Zugehörige Scripts wurden geschrieben und auf die Objekte angefügt.	Die Vorgaben konnten bis zum vereinbarten Zeitpunkt vollständig und zufriedenstellend eingehalten werden. Es traten keine Probleme bei den Ausführungen auf.

ARBEITSPAKET Nr. 2.2.1	**Schuljahr: 2014/15**
PROJEKT: **ABC-Darian**	Abteilung: Elektronik und Technische Informatik
PROJEKTPHASE: 2. Iteration **ARBEITSPAKET:** Levelauswahl	Datum: 27.11.2014 Bearbeiter: Zammernig e-mail: andreas.zammernig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel

Um die einzelnen Level aufrufen zu können, sollte eine Levelauswahl erstellt werden. Dabei sollte die Kinect eingebunden werden. Es soll möglich sein mit und ohne ihr die Levelauswahl zu steuern.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Ein Hase der sich von Wegpunkt zu Wegpunkt bewegt, stellte das Grundgerüst dar. Jeder Wegpunkt stellt einen Level dar. Entweder mit den Pfeiltasten oder mit der Kinect sollte gesteuert werden können, welcher Level ausgewählt wird. Die Scene sollte auch noch mit Umgebungsobjekten geschmückt werden, um ein authentischeres Aussehen zu verleihen. Ein Pfeil zeigt dem Hasen an, auf welchen dieser springen sollte. Der Hase sollte die ganze Zeit hüpfen und nur dann zum nächsten Wegpukt sich bewegen, wenn er unter einer gewissen Höhe ist. Per Programmcode sollte den zum Wegpunkt zugehörigen Level starten. Es muss auch sichergestellt werden, dass der richtige Level gestartet wird und dass alle Levels beinhaltet sind.	Die Vorgaben konnten bis zum vereinbarten Zeitpunkt vollständig und zufriedendenstellend eingehalten werden. Es traten keine Probleme bei den Ausführungen auf.

ARBEITSPAKET Nr. 3.2.1	**Schuljahr: 2014/15**
PROJEKT: **ABC-Darian**	Abteilung: Elektronik und Technische Informatik
PROJEKTPHASE: 3. Iteration **ARBEITSPAKET:** Buchstaben erraten	Datum: 28.01.2015 Bearbeiter: Zammernig e-mail: andreas.zammernig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel
Das Level Buchstaben erraten, sollte als nächstes realisiert werden. Hierbei geht es darum ein Wort zu erraten anhand seines Anfangsbuchstaben. Mit deiesem Level sollte auch wieder bei dem Kind das Gefühl für Wörter gegeben werden.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Mit dem Wissen über die Behandlungsmöglichkeiten von Legasthenie wurde der nächste Level entworfen. Ein zufällig ausgewähltes Wort wird in ein Textfeld geschrieben. Beim Wort selbst ist nur der erste Buchstabe zu sehen. Eine Tastatur die per Script erstellt wird, dient dazu Buchstaben in das Wort einzufüllen. Ein Turm aus mehreren Zylindern mit einem Hasen an der Spitze wurde erstellt, um den Level spannender zu machen. Ein Algorithmus stellt sicher, dass bei einem falsch ausgewählten Buchstabe der Turm kleiner wird und bei einem Buchstaben der in das Wort passt, dieser eingefügt wird.	Die Vorgabe wurde bis zum vereinbarten Datum pünktlich und richtig fertiggestellt. Es traten keine Probleme bei der Durchführung dieser Arbeit auf.

ARBEITSPAKET Nr. 4.2.1	**Schuljahr: 2014/15**
PROJEKT: **ABC-Darian**	Abteilung: Elektronik und Technische Informatik

PROJEKTPHASE:	4. Iteration	Datum:	04.03.2015
ARBEITSPAKET:	Schiffeversenken	Bearbeiter:	Zammernig
		e-mail:	andreas.zammernig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel
Als nächstes sollte ein weiteres Minispiel realisiert. Hierbei handelt es sich um eine abgewandelte Version vom originalen Spiel „Schiffeversenken“.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Es müssen zwei Spielflächen erstellt werden. Auf der einen sollte das Kind nach Wörtern suchen und auf dem anderen der Computer. Es geht darum wer zuerst alle versteckten Wörter zu finden. Ein Algorithmus stellt sicher, dass die spielende Perons gegen einen Computergegner spielen kann. Dieser wird gesteuert und sollte intelligente Züge vollstrecken können. Als Indikatoren, wie viele Wortschiffe noch gefunden werden müssen, dienen vier Gameobjekte über den Spielflächen. Die Wörter selbst werden zuerst in Arrays gespeichert, welche zufällig und mit einem Script das sicherstellt das sie gut verteilt sind, verteilt werden.	Die Vorgaben konnten bis zum vereinbarten Zeitpunkt vollständig und zufriedendenstellend eingehalten werden. Es traten keine Probleme bei den Ausführungen auf.

ARBEITSPAKET Nr. 5.2.1	**Schuljahr: 2014/15**
PROJEKT: ABC-Darian	Abteilung: Elektronik und Technische Informatik

PROJEKTPHASE:	5. Iteration	Datum:	15.04.2015
ARBEITSPAKET:	Wordshooter und Fertigstellung	Bearbeiter:	Zammernig
		e-mail:	andreas.zammernig@htl-klu.at

Verteiler:		
Zammernig Andreas	Sereinig Vanessa	Dipl.-Ing. Dr. Helmut Jeremias

Arbeitspaket-Ziel

Zeitgemäß zur letzten Iteration sollte noch ein letzter Level fertiggestellt werden und anschließend alle Level auf dasselbe Design gebracht werden und auf Fehler getestet.

Inhalt des Arbeitspaketes	**Erledigungsvermerke**
Im letzten zu erstellenden Level geht es darum, zu erkennen ob ein Wort Rechtschreibfehler enthält oder nicht. In gleichen Zeitabständen sollten Wörter auf Cubes abgebildet werden und auf die spielende Person zufliegen. Es geht darum alle falsch geschrieben Wörter anzuklicken. Bei Klicken eines richtigen Wortes erhält man Minuspunkte und bei falsch geschriebenen Wörter Punkte. Ein Timer stellt sicher, dass der Level nur 60 Sekunden lang ausgeführt wird. Es geht darum in dieser Zeit möglichst viele Punkte zu erreichen. Als Letztes wurde noch da gesamtes Spiel zusammengefügt und auf dasselbe Design gebracht. Es wurde auch sichergestellt, dass es als Gesamtprojekt funktioniert und alle Fehler beheben werde.	Die Vorgaben konnten bis zum vereinbarten Zeitpunkt vollständig und zufriedendenstellend eingehalten werden. Es traten keine Probleme bei den Ausführungen auf.

7.5 Businessplan

7.5.1 Executive Summary

7.5.1.1 Produktidee

Das Projekt ABC-Darian ist ein Computerspiel, welches Kindern und Jugendlichen, die an Legasthenie oder Dyskalkulie leiden, helfen soll, ihre Lernschwäche abzubauen. Es soll eine spielerische Ergänzung zu den herkömmlichen Therapien und Unterrichtsstunden sein und den Kindern zeigen, dass solche Übungen auch Spaß machen können.

7.5.1.2 Marktpotential

Die Innovation von ABC-Darian steckt darin, dass nicht nur Lesen, Schreiben und Rechnen trainiert, sondern ebenso das räumliche Denkvermögen durch die Arbeit im dreidimensionalen Raum geschult wird. Zusätzlich wird auch noch eine neuartige Form der Spielsteuerung – die Gestensteuerung mittels Kinect – in das Spielgeschehen eingebaut.

Vergleichbare Anbieter existieren nur auf Websites, auf denen man online Minispiele im 2D-Bereich spielen kann. Ein Produkt wie ABC-Darian, das auch noch ohne Internetzugang benutzt werden kann, ist auf dem Markt bisher noch nicht vertreten.

In Kärnten leiden ca. 8.500 Kinder an einer Form von Legasthenie.

7.5.1.3 Finanzierung

Die Selbstkosten belaufen sich auf 9,13€/Stück. Der Long-Term Bottom Price wird auf 30€/Stück festgelegt. Fixed Assets müssen zu Beginn nicht getätigt werden, da sowohl Laptops als auch die Kinect aus dem Privatvermögen eingebracht wurden. Der Kapitalbedarf beläuft sich daher lediglich auf die Abschreibungen der Hardware, auf anfallende Neuanschaffungen auf Grund der Wertverluste der Geräte und auf eventuelle Werbekampagnen.

Geplant ist ein jährlicher Umsatz von 29640€ und ein Gewinn von etwa 28.000€, wenn man von einer Verkaufszahl von 988 verkauften Exemplaren pro Jahr ausgeht.

7.5.1.4 Geschäftsstrategie

Es wird die Rechtsform Offene Gesellschaft(OG) gewählt, da die Gründungskosten, wie der Eintrag ins Firmenbuch und die Gewerbeanmeldung, gering sind.

Das Produkt wird an Schulen zur Unterrichtserweiterung von Sonderkursen für lernschwache Kinder angeboten und an Privatpersonen verkauft. Da das Angebot in Bildungseinrichtungen an Werbung nicht ausreicht, werden zusätzlich Folder angefertigt und Anzeigen in lokalen Zeitungen gedruckt. Für den direkten Kundenkontakt ist außerdem das Betreiben von Messeständen sinnvoll.

7.5.2 Ist-Analyse (Produkt, Umfeld, Unternehmen, Team)

7.5.2.1 Zielgruppe

Laut der Seite dyslexiaserver.com, welche von **Statistics Austria** ihre Informationen erhält, gibt es in Kärnten 8.628 Kinder, die an Legasthenie bzw. Formen davon (Dyskalkulie) leiden. Dies entspricht einem Prozentsatz von 15% in Bezug auf die 56.850 in Kärnten lebenden Kinder (Zahlen: Stand 2013).

Die Werte auf ganz Österreich bezogen sehen wie folgt aus: 129.759 von 865.060 Kindern leiden an einer Lernschwäche, womit sich auch österreichweit ein Prozentsatz von 15% ergibt. [STAT 15]

7.5.2.2 Konkurrenz

„LegaKids“

- Bieten spielerisch Training für legasthene und dyskalkule Kinder an
- Die Stiftung nimmt Spenden an
- Bietet Spiele gegen Mitgliedschaft gratis im Internet an
- Großteil der Spiele wird nur online im Browser zur Verfügung gestellt [LEGA15]

Differenzierung zu der Konkurrenz (USPs)

- ABC-Darian wird zum Download bzw. auf Sticks/CDs angeboten, was das Spielen ohne Internet möglich macht
- Arbeit im 3D-Bereich und bessere Grafik
- Die Gestensteuerung von ABC-Darian ist innovativ

7.5.2.3 SWOT-Analyse

Mit dem Instrument der SWOT-Analyse werden sowohl innerbetriebliche Stärken und Schwächen (Strenghts – Weaknesses), als auch externe Chancen und Gefahren (Opportunities – Threats) analysiert.

In der internen Analyse werden die Stärken bzw. Schwächen untersucht, für die das Unternehmen selbst verantwortlich ist.

Interne Analyse	
Stärken(Strengths)	**Schwächen(Weaknesses)**
Innovative Umsetzung der Spielideen (3D, KINECT,…)	Geringer Bekanntheitsgrad
Kostenvorteile (keine Lagerkosten,…)	Gering ausgebauter Betrieb
Gute Vertriebskanäle (Schulen)	
Größtenteils Unabhängigkeit von anderen Betrieben	
Software braucht keine Updates	

Bei der externen Analyse wird die Unternehmensumwelt untersucht. Die Chancen bzw. Risiken kommen von außen und ergeben sich aus Veränderungen im Markt.

Externe Analyse	
Chancen(Opportunities)	**Risiken(Threats)**
Förderungen	Bei Kunden wird teils Kinect vorausgesetzt
Hohes Marktpotential	Steigende Konkurrenz
Schwache Konkurrenz	Zu schneller Heilungsprozess
Bekanntheitsgradsteigerung durch Wettbewerbe	

SWOT-Analyse		Interne Analyse	
		Stärken (Strengths)	**Schwächen(Weaknesses)**
Externe Analyse	**Chancen (Opportunities)**	**S-O-Strategie:**	**W-O-Strategie**
		Da wir einen enormen Kostenvorteil durch die geringen Betriebsmittelkosten haben, ist es uns möglich Anlagen auf andere Weise zu investieren. Durch den Zukauf von Werbeplakaten können wir in unserem ohnehin konkurrenzschwachen Markt an Bekanntheit gewinnen.	Dank der guten Qualität und innovativen Umsetzung unseres Spieles, können wir durch die Teilnahme an Wettbewerben und mögl. Gewinnen dieser die Kunden auf uns aufmerksam machen. Somit kann unser geringer Bekanntheitsgrad verbessert werden.
	Gefahren (Threats)	**S-T-Strategie**	**W-T-Strategie**

		Damit die Kinect nicht für das gesamte Spiel vorausgesetzt werden muss, wird es auch Level geben, in denen sie nicht von Nöten sein wird. So können Kunden unser Produkt nutzen, ohne sich durch die Anschaffung einer Kinect abgeschreckt zu fühlen.	Durch den geringen Bekanntheitsgrad, den wir zu Beginn haben werden, wäre eine Steigerung der Konkurrenz in unserem Markt eine fatale Schwächung unseres Verkaufspotentials. Deshalb kommt uns unser guter Ruf an Schulen zu Gute.

7.5.2.4 Rechtsform

Zu Beginn der Unternehmung wird die Rechtsform „Offene Gesellschaft(OG)“ gewählt, da anfänglich das Mindeststammkapital und die Gebühren für den Notar nicht aufgebracht werden können.

Vorteil:

- Geringe Steuerbelastung

  Da der auf die Gesellschafter aufgeteilte Gewinn kleiner als 11000€ ist. Kommt dem Projekt zu Gute, da der eventuelle Gewinn von Fördergeldern somit für die Anschaffung von Werbungsmaterial und Speichermedien (CDs, Sticks,…) zum Verkauf ausgeben werden kann. Eine Mindestkörperschaftssteuer würde dem Unternehmen schaden, da zu Beginn noch keine Rücklagen gebildet werden können.

Auch in weiterer Folge wird diese Rechtsform beibehalten, da die Kosten für die Unternehmungsgründung einer GmbH zu hohe Ausgaben darstellen. Da ABC-Darian keine laufenden Fixkosten verursacht, ist auch das Risiko durch die Haftung der Gesellschafter sehr gering.

7.5.2.4.1 Gesellschafter

Gesellschafter der OG sind Andreas Zammernig und Vanessa Sereinig. Inoffizielle Arbeitsaufteilung:

Die Mitentwicklung am Programm sowie Vorstellung des Produkts an Schulen wird von beiden Gesellschaftern erledigt.

- Zammernig:
  - Buchführung
- Sereinig:
  - Marketing (Werbeaufträge veranlassen,…)

7.5.3 Marketingplan

7.5.3.1 Product (Produkt- und Leistungspolitik)

Produktebene	Kundennutzen	USP
Kernprodukt: Computerspiel	**Kernnutzen:** Durch Kauf und Verwendung des Spiels können Kinder und Jugendliche ihre Lernschwäche abbauen.	**Produktinnovation:** Lernspiel für Kinder mit Lernschwäche, welches im dreidimensionalen Bereich arbeitet. Auch die Einbindung der Kinect ist einzigartig.
formales Produkt: Online-Version auf Website erweitertes Produkt: gratis Angebot für Schulen, was auch als Werbung dient	**Zusatznutzen:** Das Spiel kann online ausprobiert werden Updates der Software sind möglich, werden auch nicht gebraucht	**Differenzierung:** Das Spiel wird sowohl mit der Zusatzfunktion der Kinect angeboten, als auch als Light-Version für Kunden die die Kinect nicht besitzen.

- Produktvariation für verschiedene Kundensegmente:

Die Schwierigkeitsgrade sind variabel, so können sowohl Jugendliche als auch Kinder das Spiel nutzen.

Variation ist für zwei Segmente gegeben:

1. Schulen:
   Können das Spiel (oder Teile davon) gratis nutzen, dafür wird die Software in der Funktionalität gekürzt (bzw. Anzahl der Level). Dies ist ebenfalls eine Art Werbung für das Produkt.

2. Privatpersonen:
   Kaufen das Spiel für die Benutzung zu Hause. Vollständige Versionen mit und ohne Kinect (kann in den Einstellungen beschlossen werden).

7.5.3.2 Price (Preis- und Konditionenpolitik)

7.5.3.2.1 Kostenorientierte Preisbildung

- Kalkulation der Selbstkosten

  - Siehe
- fixe und variable Kosten
  - Fixkosten: Strom- und Internetrechnung
  - Variable Kosten: Speichermedien zum Produktvertrieb
- Personalkosten
  - Keine Personalkosten, Gewinnausschüttung lediglich an die zwei Gesellschafter
- Fremdleistungskosten
  - Eventuell anfallende Kosten für die Website und das Erstellen der Speichermedien für den Vertrieb des Produktes
- Nachfrageorientierte Preisbildung (elastisch / unelastisch)
  - Das Produkt ist unelastisch, da das Wohlergehen von Kindern im Vordergrund steht und da es zur Bekämpfung einer Lernschwäch entwickelt wurde. Auch bei Preiserhöhungen werden die Eltern das Produkt für den Lernerfolg ihres Nachwuchses kaufen.

7.5.3.2.2 Konkurrenzorientierte Preisbildung

Da es nahezu keine Konkurrenz im Bereich der Legasthenie- und Dyskalkuliebekämpfung gibt, ist der preisliche Druck sehr gering. Dies ermöglicht uns die freie Wahl der Preisstrategie.

- Gewählte Methode:
  - Skimming- oder Abschöpfungmethode:
    Das Produkt ist sehr innovativ und wirkt daher sehr ansprechend auf Kunden.
- Konkurrenz:
  - Die Konkurrenz bietet vorwiegend gratis im Internet an (auf Spenden basiert).
- Long-Term Botton Price:
  - Beläuft sich auf 30€/Stück
  - Kalkulation siehe

7.5.3.3 Placement (Distribution-/Vertriebspolitik)

Merkmale der Vertriebspolitik:

- Direkter Vertriebsweg
- Vertriebsart
  - Verkauf über das Internet,
  - D-Verkauf auf Messen
  - Verkauf des Spiels als App im AppStore bzw. PlayStore

- Vorteile und Gründe
  - Direkter Vertrieb ermöglicht es , das Produkt zu höheren Preisen zu verkaufen
  - Bessere Beziehung zu Kunden
  - Der Kunde kann durch das Gespräch mit den Entwicklern direkt professionell beraten werden
  - Unabhängigkeit

- Downloads sind heutzutage wegen der Bequemlichkeit meist gefragter, hierzu wird keine Vertriebsstelle benötigt
- Einsparung diverser anfallender Kosten, um das Produkt bei Vertriebsstellen anbieten zu können (Provisionen abhängig von Umsatz)
- Zusätzliches Personal zur Erläuterung der Produktidee (z.B. Reisekosten der Promoter) wird vorerst wegen des fachlichen Wissens der Gesellschafter und den Werbefoldern nicht benötigt. Später allerdings könnten diese zusätzlichen Kosten aufgebracht werden müssen.

7.5.3.4 Promotion (Kommunikationspolitik)

Es wird einerseits auf die klassische Werbung zurückgegriffen, und andererseits auf den persönlichen Verkauf.

- Klassische Werbung:

  - Werbeprospekte wie Folder und Flyer(werden vor allen in Schulen ausgehändigt)
  - kleine Anzeigen in der lokalen Zeitung
  - Werbung auf Internetseiten (sehr günstig!)

- Persönlicher Verkauf:

  - Produkt wird auf Messeständen vorgeführt und dort auch gleich auf Hardware (USB-Sticks, CDs,…) zum Verkauf angeboten

7.5.3.4.1 Tausender Kontaktpreise für verschiedene Werbemittel

Werbetyp: Folder

Laut Statistik Österreich besuchen ca. 20.000 Kinder eine Volksschule in Kärnten.

Außerdem wissen wir, dass in Kärnten ca. 8.500 Kinder an einer Form von Legasthenie leiden. Wir werden also eine Auflage von rund 10.000 Folder benötigen, die wir an die 236 Volksschulen in Kärnten verteilen.

Laut der angegebenen Webadresse kosten 4-seitige Flyer im normalen Design zu 10.000 Stück in etwa 200€ (inkl. Lieferung und Anfertigung). Die Reichweite beträgt in dem Fall mindestens die Anzahl der gefertigten Folder. [FOLD15]

$$Tausender\ \text{Kontaktpreis} = \frac{Werbekosten}{ReicHweite} * 1.000 =$$

$$= \frac{200}{10.000} * 1.000 = \frac{20€}{1.000\ Personen}$$

Beim Werbetyp Folder 1000 Personen zu erreichen, betragen die Kosten 20€.

Werbetyp: Lokale Zeitung

Die Kosten für eine Fotoanzeige in der Kärntner und Steirer Ausgabe der Kleinen Zeitung über eine Woche betragen durchschnittlich 82,50€ Die Zeitung hat in den

beiden Bundesländern im Halbjahr 2014 circa 1499 Abonnements (https://www.kleinezeitung.at/k/service/tarif/3879075/Die-aktuellen-Zahlen-der-Osterreichischen-Auflagenkontrolle). Die Einzelverkäufe werden bei der Berechnung nicht berücksichtigt. [LOKZ15]

$$\text{Tausender Kontaktpreis} = \frac{Werbekosten}{ReicHweite} * 1.000 = \\ = \frac{82{,}50€}{1490} * 1.000 = \sim \frac{0{,}0554€}{1.000\ Personen}$$

Beim Werbetyp lokale Zeitung 1000 Personen zu erreichen, betragen die Kosten 0,06€.

Werbetyp: Messestand

Durchschnittlich beträgt die Besucheranzahl bei Gesundheitsmessen 10.000 Besucher. Eine solche Messe ideal ist um ein Programm zur Bekämpfung von Legasthenie und Dyskalkulie vorzustellen. Für einen Kleinbetrieb würde ein Messestand mit Tisch, Stühlen und einem Teppich (699€ + 108€ + 72€) würde man dann 879€ kosten. Bei einem solchen Stand kann sehr gut beraten und Fragen gestellt werden, auch könnte das Produkt direkt vor Ort am Stand vertrieben werden. [MES]

$$\text{Tausender Kontaktpreis} = \frac{Werbekosten}{ReicHweite} * 1.000 = \\ = \frac{879€}{10.000} * 1.000 = \frac{87€}{1.000\ Personen}$$

Beim Werbetyp Messestand 1000 Personen zu erreichen, betragen die Kosten 87€.

Fazit

Die Reichweite ist bei den Werbetyp Messestand und Folder am größten. Die Kosten sind bei dem Messestand am höher, dafür wäre auch eine Beratung vor Ort möglich. Der Werbetyp lokale Zeitung wäre die billigste Variante, hat dafür eine geringere Reichweite.

7.5.3.4.2 Marketing Mix

Die grundsätzliche Strategie wird die **langsame Marktabschöpfung** sein.

- Vorteile
  - Es gibt wenig bis keine unmittelbare Konkurrenz
  - Käufer sind bereit, einen hohen Preis zu bezahlen, da es um das Schulen der geistigen Fähigkeiten ihrer Kinder geht
  - Die fehlende Konkurrenz lässt viel Spielraum in der Preissetzung
  - Durch die geringen Produktions- und Personalkosten ist es uns möglich relativ viel Geld in die Kommunikationspolitik zu investieren.
- Nachteil

- Das Produkt ist nicht bekannt (Kann durch die hervorragende Kommunikationspolitik aufgewogen werden: Das kostenfreie Angebot an Schulen dient der optimalen Mundpropaganda).

7.5.4 Projektplan

Meilenstein	Zeitraum															
	2014								2015							
	Mai	Juni	Juli	August	September	Oktober	November	Dezember	Jänner	Februar	März	April	Mai	Juni	Juli	August
Erstellung Geschäftsmodell/Businessplan																
Entwicklung des Prototypen																
Optimierung/Verbesserung																
Gründung des Unternehmens																
Investitionsplanung und Kapitalbeschaffung																
Materialbeschaffung																
Auftragseingänge																

Die Entwicklung der Projektidee und des Prototypen erfolgte als erstes, im Mai 2014, und dauert auch am längsten. Es ist somit das größte aller Arbeitspakete.
Am Anfang musste überlegt werden, wie die Idee umgesetzt werden soll. Sobald dies festgelegt war, konnte mit der Arbeit begonnen werden.
Die Materialbeschaffung gleich nach der Festsetzung der Grundidee und der ersten Pläne statt. Es mussten die Kinect und die Programme angeschafft werden, welche für die Umsetzung von Beginn an nötig waren. Während des weiteren Verlaufs des Programms sind keine Beschaffungen von Material mehr nötig. Höchstens wenn Reparaturen oder Neuanschaffungen von Geräten notwendig sind.
Sobald erste Prototypen in Testversionen vorhanden sind und das sich das Projekt dem Ziel immer mehr nähert, beginnt die Erstellung des Businessplans. Da dieser unbedingt vor der Unternehmensgründung fertig sein muss, wurde schon im Oktober 2014 begonnen, sodass wir vier Monate parallel zur Entwicklung dazu Zeit haben.
Bei auftretenden Fehlern müssen parallel zur Entwicklung zusätzlich Verbesserungen beziehungsweise Optimierungen am Projekt vorgenommen werden, um das Produkt rechtzeitig auf den Markt zu bringen. Diese Änderungen und Behebungen sind also im letzten Abschnitt der Prototypenentwicklung zu beginnen und bis zur Unternehmungsgründung durchzuführen, um bis dahin einen vollständig funktionierenden Prototypen entwickelt zu haben.
Die Gründung des Unternehmens selbst kann erst dann erfolgen, wenn eine Version des Projektes vorhanden ist (wenn der erste Prototyp funktioniert und zum Verkauf angeboten wird). Sie erfolgt de dem nach erst im Juni 2015.
Die Investitionsplanung begann im Jänner 2015 mit der Überlegung, wie das Projekt beworben wird und mit welchen Mittel die Werbung finanziert wird. Die Kapitalbeschaffung fällt sehr gering aus, da keine hohe Kosten im Zuge des Projekts anfallen.

7.5.5 Finanzplan (1. Geschäftsjahr)

7.5.5.1 P&L-Account

1. Jahr (ab Juni 2015)			
	Art/Bezeichnung	**€/Monat**	**€/Jahr bzw. 7 Monate)**
Einnahmen			17290€/Jahr
Ausgaben	Werbung	82,50€/Woche	-165 (2 Wochen)
	Telefon und Internet	25	-175
	Depreciation Laptops (2Stk)	33,3	-400 (für 2 Stk)
	Low cost assets	-400 (im 1. Jahr)	
Betriebsergebnis			16350
Ausschüttung			-16350
Einlagen			16350

7.5.5.2Stundensatzkalkulation

Stundensatzkalkulation			Mitarbeiter bzw. meine eigene Arbeitsleistung / Stunde --> muss dem Kunde		
Bruttogehalt	**2000**			%	
LNK (SV Abgaben)	620,6			21,60	DG Beitrag SV
Personalkosten	0			4,50	DG Beitrag Familienlastenausgleichsfonds
inkl. SV Abgaben	2620,6			0,40	Zuschlag zu DG Beitrag
Sonderzahlungen (SZ)				3,00	Kommunalsteuer
13./ 14 Gehalt	333,333333			1,53	Beitrag zur Mitarbeitervorsorgekasse
				31,03	**SV Abgaben**
Personalkosten inkl					
SV Beitrag und SZ	**2953,93333**			**SZ**	131,03 *2 Gehälter / 12 Monat
LNK inkl. SV und SZ	**rd. 53%**				
Anwesenheitszeit	260	Tage			52 Wochen * 5 Tage
abzügl. Urlaub	25				
abzügl. Feiertage	13				
abzügl. Krankenstand	13				
abzügl.sonst. Verhinderunge	5				
Anwesenheitszeit	**204**				
Stunden Anwesenheitszeit	**204 *7,7 =**	**1571 Stunden**		*	38,5 Stundenwoche / 5 Tage = 7,7
2954€ pro Monat	*12 Monate	/1571 Stunden	22,56€/Stunde		

7.5.5.3 Berechnungen

$$gesamte\ Entwicklungszeit\ des\ Projekts: 400h$$

$$Sales\ volume = 19\ \frac{Stk}{Woche} = 988\frac{Stk}{Jahr}$$

$$Sales\ revenues = \ 30€ * 988\frac{Stk.}{Jahr} = 29640\frac{€}{Jahr} = 17290€\ im\ 1.Jahr\ (ab\ Juni\ 2015)$$

$$Cost\ of\ sales\ \left(\frac{Produktionskosten}{Stk}\right) = \frac{Ausgaben}{Sales\ volume} = \frac{9024€}{988Stk} = \frac{9{,}13€}{Stk}$$

$$Entwicklungskosten = Stundensatz * insgesamter\ Arbeitsaufwand\ in\ Stunden(h)$$
$$= 22{,}56\frac{€}{h} * 400h = 9024€$$

$$long\ term\ bottom\ price = 30€$$

7.5.5.4 Eröffnungs- und Schlussbilanz

Opening Balance Sheet (1.6.2015)			
Aktiva		**Passiva**	
Anlagevermögen		***Eigenkapital***	
Laptop	600€	Eigenkapital	1000€
Aktuelles Vermögen			
Liquide Mittel	400€		
Gesamtvermögen	**1000€**	**Gesamtkapital**	**1000€**

Closing Balance Sheet (31.12.2015)			
Aktiva		**Passiva**	
Anlagevermögen		***Eigenkapital***	
Laptop	400€	Eigenkapital	17350€
Aktuelles Vermögen			
Liquide Mittel	16950€		
Gesamtvermögen	**17350€**	**Gesamtkapital**	**17350€**

7.5.5.5 Liquiditätsplan

	heute	in 1 Monat	in 2 Monaten	in 3 Monaten	in 4 Monaten	in 5 Monaten	in 6 Monaten	in 7 Monaten
Kontostand ->	1 000,00	2 880,00	5 325,00	7 770,00	10 215,00	12 660,00	15 105,00	17 350,00
Einnahmen ->		2 470,00	2 470,00	2 470,00	2 470,00	2 470,00	2 470,00	2 470,00
Ausgaben:								
Werbung ->		165,00						
Firmenhandy ->		200,00						
Kinect ->		200,00						
Telefon / Internet ->		25,00	25,00	25,00	25,00	25,00	25,00	25,00
Depreciation ->								200,00

7.5.5.6 Break Even Point (BEP)

- BEP ohne Entwicklungskosten:

$$break\ even\ point = \frac{Fixe\ Kosten}{Verkaufspreis - variable\ Kosten} = \frac{25€ * 7 + 600€}{30€ - 0{,}2€} = \sim 26€$$

- BEP mit Entwicklungskosten:

$$break\ even\ point = \frac{Fixe\ Kosten + Entwicklungskosten}{Verkaufspreis - variable\ Kosten} = \frac{25€ * 7 + 600€ + 9024€}{30€ - 0{,}2€} =$$

$$= \sim 328{,}83€$$

7.5.6 Business Model Canvas

Key Partners	Key Activities	Value Proposition	Customer Relationships	Customer Segments
Hersteller Hersteller für die Produktion/zur Zurverfügungstellung von der nötigen Hardware gebraucht (CDs, USB-Sticks,...) **Website (?)** Entwickler der Website, auf der die Software heruntergeladen werden kann **Marketingpartner** Schulen, Ministerium, Landesschulrat, Wettbewerbe **Werbung** Hersteller von Plakaten und Broschüren	**Programmierung** Erstellen der Software, Gestensteuerung (Durch Einbindung der KINECT können Gesten erfasst und im Spiel verarbeitet werden) **Buchhaltung** Kostenrechnung **Key Resources** **Hardware** für die Speicherung der Software (CDs, USB-Sticks,...) **Website** zum Download der Software **Computerequipment** PC, Maus, KINECT und Tastatur werden zur Entwicklung des Projektes benötigt **Programme** Die notwendigen Programme sind gratis	**Lernschwächen bekämpfen** Abbau von Legasthenie und Dyskalkulie in spielerischer Umgebung **Räumliches Denkvermögen steigern** Mittels Unity3D und BLENDER wird eine dreidimensionale Umgebung erschaffen **Systemvoraussetzung** Für einige Level wird die KINECT vorausgesetzt	**Förderung von Bildungseinrichtungen** Kostenfreie Zurverfügungstellung der Lernsoftware für Schulen mit Förderklassen/hohem Anteil an lernschwachen Kindern **Handbuch** Erläuterung zur Verwendung der Software auf der Website **Channels** **Messen** Präsentation des eigenen Produktes mittels Messeständen **Websites und CDs/Sticks** Vertrieb der Software durch Download oder Erwerb auf Hardware **Schulen** Vorstellung des Produktes an Volksschulen	**Familien** Familien, in denen Kinder unter einer Lernschwäche leiden **Bildungseinrichtungen** Kinder und Jugendliche im Volks- und Unterstufenalter (in Österreich)
Cost Structure **Werbung** Kosten zur Anbringung eines Plakates oder Werbebanner auf Websites, Broschüren für Präsentationen **Website** Erstellungs- und Erhaltungskosten **Hardware** Einkauf der nötigen Hardware zum Vertrieb		**Revenue Streams** **Vertrieb** Einnahmen bei Verkauf der Software **Asset Store** Teile des Programmes können im Asset Store von Unity3D gekauft werden		

7.5.7 Literaturverzeichnis

[LEGA15] www.legakids.net, Stand 2015

[FOLD15] „Anbieter Folder“ http://www.cewe-print.at/folder-jetzt-bestellen.aspx, Stand 2015

[LOKZ15] „Anbieter lokale“ Zeitung:http://www.kleinezeitung.at/s/service/tarif/inserieren/index.do, Stand 2015

[MESS15] „Anbieter Messestand“, http://www.cmw.at/pdf/719.pdf), Stand 2015

[STAT 15] Statistics Austria, http://www.dyslexiaserver.com/statistik/index.php?action=K%E4rnten, Stand 2015

7.6 Abbildungsverzeichnis

Abbildung 3-1: Die AFS-Methode 12
Abbildung 3-2: Aufbau eines Skripts in Unity3D....16
Abbildung 4-1: Blender Logo 22
Abbildung 4-2: Big Bucks Bunny 23
Abbildung 4-3: Einsames Haus mit Rendering 23
Abbildung 4-4: Transormation zweier Metabälle 24
Abbildung 4-5: Modemenü in Blender....26
Abbildung 4-6: Tools im Edit-Mode....27
Abbildung 4-7: Beispiel für File IO 30
Abbildung 4-8: Scene des Instruktionsbildschirms....32
Abbildung 4-9: Szenenobjekte des Logos....35
Abbildung 4-10: Logo in anderer Ansicht....35
Abbildung 4-11: World-Einstellungen des Logos 36
Abbildung 4-12: Logo 36
Abbildung 4-13: World-Einstellungen des Banners....37
Abbildung 4-14: Banner....37
Abbildung 4-15: Apfel im Edit-Mode 38
Abbildung 4-16: 3D Text-Mesh im Edit-Mode 39
Abbildung 4-17: Buchstabe im Object-Mode....39
Abbildung 4-18: Buchstabenkarte im Object-Mode....40
Abbildung 4-19: Labyrinth-Objekt im Edit Mode....41
Abbildung 4-20: Fertige Schüssel....41
Abbildung 4-21: Schüssel im Edit-Mode 41
Abbildung 4-22: Assets von "Zahlen einsammeln"....42
Abbildung 4-23: Skripts in Memory....44
Abbildung 4-24: Rechnungstextur 44
Abbildung 4-25: Ergebnistextur 44
Abbildung 4-26: Vergleich der Kameraeinstellungen (links: Minimap)....48
Abbildung 4-27: Highscore-Liste vom Spiel "Futter sammeln" 52
Abbildung 4-28: Startmenü....52
Abbildung 4-29: Namen eingeben 53

Abbildung 4-30: Spiel beenden....53
Abbildung 4-31: Instruktionsbildschirm (Bsp: Memory)....54
Abbildung 4-32: Spieleansicht von „Äpfel zählen“....55
Abbildung 4-33: Spieleansicht "Würfel zählen"....56
Abbildung 4-34: Spieleansicht von „Zahlen einsammeln“....56
Abbildung 4-35: Spieleansicht des Spiels „Memory“....57
Abbildung 4-36: Kartenanzahl auswählen....57
Abbildung 4-37: Spieleansicht "Wrong Or Right"....58
Abbildung 4-38: Spieleansicht "Futter einsammeln"....59
Abbildung 4-39: Spieleansicht "Labyrinth"....59
Abbildung 4-40: Minimap im Detail....60
Abbildung 4-41: Beispiel für Legasthenie [BSP 15]....63
Abbildung 4-42: Blockschaltbild OSIT-Modell [OSI 15]....67
Abbildung 4-43 Kinect [KIN 15]....68
Abbildung 4-44: Interne Verschaltung Kinect [BLS 15]....70
Abbildung 4-45: Voransicht der Levelauswahl....75
Abbildung 4-46: Pfeil für Levelauswahl....77
Abbildung 4-47: Modell des Hasen und Turmes....78
Abbildung 4-48: Modell der Tastatur....80
Abbildung 4-49: Spielansicht der Levelauswahl....81
Abbildung 4-50: Spielansicht des Shooters....82
Abbildung 4-51: Ende des Shooter-Levels....82
Abbildung 4-52: Spielansicht des Wörter erraten....83
Abbildung 4-53: Wörter erraten - Spiel gewonnen....83
Abbildung 4-54: Wörter erraten - Spiel verloren....84
Abbildung 4-55: Spielansicht des Schiffeversenken....84
Abbildung 4-56: Voransicht des Buchstaben einsammeln....85
Abbildung 4-57: Spielansicht des Buchstaben einsammeln....85
Abbildung 7-1: Systemstruktur....93

Abbildung 7-2: Projektstruktur....94
Abbildung 7-3: Terminplan in MS Project....96
Abbildung 7-4: Projektbalkenplan....96
Abbildung 7-5: Versionen von Unity3D....99

Tabelle 4-1: Liste der essentiellen Shortcuts und Hotkeys in Blender....29
Tabelle 4-2: Funktionen von PlayerPrefs....31
Tabelle 7-1: Arbeitszeitnachweis Sereinig....97
Tabelle 7-2: Arbeitszeitennachweis Zammernig....98
Tabelle 7-3: Links von Unity....99
Tabelle 7-4: Links von Blender....100
Tabelle 7-5: Links für Kinect SDK 1.8....101
Tabelle 7-6: Link für Kinect Developer Toolkit SDK 1.8....102